Intel® Integrated Performance Primitives for Intel® Architecture

Quick Reference

Part 3: Small Matrices

Document Number: 307152-002

World Wide Web: http://developer.intel.com

Version	Version Information	Date
-001	Original issue. Documents API included into Intel IPP release 5.0. Links to Intel IPP Reference Manual (document number A68761-007).	03/2005
-002	Documents API included into Intel IPP release 5.1 gold. Links to Intel IPP Reference Manual (document number A68761-008).	02/2006

Overview

This Quick Reference provides a full list of prototypes for all functions included into the small matrices domain of the Intel® Integrated Performance Primitives (Intel® IPP) for Intel® architecture.

The Intel IPP software is a new generation of the Intel® Performance Libraries, comprising a broad range of functions for basic software functionality and including, among many others, the small matrix functions domain.

Intel IPP is a cross-architecture software library optimized for the latest generations of Intel microprocessors, including Intel® Pentium® 4 processor, Intel® Pentium® 4 processor with Streaming SIMD Extensions 3 (SSE3), Intel® Xeon® processor, Intel® Xeon® processor with Intel® Extended Memory 64 Technology (Intel® EM64T), Intel® Itanium® 2 processor, and Intel® PCA application processors.

The small matrices subset of Intel IPP includes the following functional domains:

- Utility Functions
- Vector Algebra Functions
- Matrix Algebra Functions
- Linear System Solution Functions
- Least Squares Problem Functions
- Eigenvalue Problem Functions.

This Quick Reference contains an alphabetic list of all small matrices function names with hyperlinks to the full set of function prototypes available in the library and a short description of function operation. The detailed information about all functions, including introductory material,

general argument description, function behavior explanation, return values and more, is given in the Intel® Integrated Performance Primitives Reference Manual, volume 3 (document number A68761-008).

Each section and function name included into the reference section of this Quick Reference has a hyperlink to the respective section of the above Reference Manual.

For these links to work properly, please make sure that:

1. You have the correct version of the Intel IPP Reference Manual available with the Intel IPP distribution (file name ippmman.pdf)
2. Files for the Quick Reference and for the Reference Manual reside in the same directory.

If you have jumped to the manual and wish to return to the Quick Reference, use the Go to the Previous View button in the Adobe Acrobat* tool. If you plan to jump between the Quick Reference and the Manual frequently, it is preferable to open the Reference Manual before clicking hyperlinks in the Quick Reference. In this case you will see both documents in the Window menu of your viewer.

For more information about the Intel IPP library, please refer to the Web site at developer.intel.com/software/products/ipp/.

Alphabetic List of Routines

Utility Functions

Copy

Performs copy operation.

Case 1: Vector array operation

```
IppStatus ippmCopy_va_32f_SS(const Ipp32f* pSrc, int srcStride0,
    int srcStride2, Ipp32f* pDst, int dstStride0, int dstStride2, int
    len, int count);

IppStatus ippmCopy_va_64f_SS(const Ipp64f* pSrc, int srcStride0,
    int srcStride2, Ipp64f* pDst, int dstStride0, int dstStride2, int
    len, int count);

IppStatus ippmCopy_va_32f_SP(const Ipp32f* pSrc, int srcStride0,
    int srcStride2, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmCopy_va_64f_SP(const Ipp64f* pSrc, int srcStride0,
    int srcStride2, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmCopy_va_32f_SL(const Ipp32f* pSrc, int srcStride0,
    int srcStride2, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);

IppStatus ippmCopy_va_64f_SL(const Ipp64f* pSrc, int srcStride0,
    int srcStride2, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);

IppStatus ippmCopy_va_32f_LS(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp32f* pDst, int dstStride0, int dstStride2, int
    len, int count);

IppStatus ippmCopy_va_64f_LS(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp64f* pDst, int dstStride0, int dstStride2, int
    len, int count);

IppStatus ippmCopy_va_32f_PS(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f* pDst, int dstStride0, int dstStride2, int
    len, int count);

IppStatus ippmCopy_va_64f_PS(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f* pDst, int dstStride0, int dstStride2, int
    len, int count);

IppStatus ippmCopy_va_32f_LP(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmCopy_va_64f_LP(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);
```

```
IppStatus ippmCopy_va_32f_LL(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride2, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int
   len, int count);
IppStatus ippmCopy_va_64f_LL(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride2, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int
   len, int count);
IppStatus ippmCopy_va_32f_PP(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
   len, int count);
IppStatus ippmCopy_va_64f_PP(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp64f** pDst, int dstRoiShift, int dstStride0, int
   len, int count);
IppStatus ippmCopy_va_32f_PL(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int
   len, int count);
IppStatus ippmCopy_va_64f_PL(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int
   len, int count);
```

Case 2: Matrix array operation

```
IppStatus ippmCopy_ma_32f_SS(const Ipp32f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp32f* pDst, int dstStride0,
   int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_64f_SS(const Ipp64f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp64f* pDst, int dstStride0,
   int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_32f_SP(const Ipp32f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp32f** ppDst, int dstRoiShift,
   int dstStride0, int width, int height, int count);
IppStatus ippmCopy_ma_64f_SP(const Ipp64f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp64f** ppDst, int dstRoiShift,
   int dstStride0, int width, int height, int count);
IppStatus ippmCopy_ma_32f_SL(const Ipp32f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp32f** ppDst, int dstRoiShift,
   int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_64f_SL(const Ipp64f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp64f** ppDst, int dstRoiShift,
   int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_32f_LS(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride1, int srcStride2, Ipp32f* pDst, int dstStride0,
   int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_64f_LS(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride1, int srcStride2, Ipp64f* pDst, int dstStride0,
   int dstStride1, int dstStride2, int width, int height, int count);
```

```
IppStatus ippmCopy_ma_32f_PS(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_64f_PS(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_32f_LP(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, Ipp32f** ppDst, int dstRoiShift,
    int dstStride0, int width, int height, int count);
IppStatus ippmCopy_ma_64f_LP(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, Ipp64f** ppDst, int dstRoiShift,
    int dstStride0, int width, int height, int count);
IppStatus ippmCopy_ma_32f_LL(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, Ipp32f** ppDst, int dstRoiShift,
    int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_64f_LL(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, Ipp64f** ppDst, int dstRoiShift,
    int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_32f_PP(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);
IppStatus ippmCopy_ma_64f_PP(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);
IppStatus ippmCopy_ma_32f_PL(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int width, int height, int count);
IppStatus ippmCopy_ma_64f_PL(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int width, int height, int count);
```

Extract

Performs ROI extraction.

Case 1: Vector operation

```
IppStatus ippmExtract_v_32f(const Ipp32f* pSrc, int srcStride2, Ipp32f*
    pDst, int len);
IppStatus ippmExtract_v_64f(const Ipp64f* pSrc, int srcStride2, Ipp64f*
    pDst, int len);
IppStatus ippmExtract_v_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    Ipp32f* pDst, int len);
IppStatus ippmExtract_v_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    Ipp64f* pDst, int len);
```

Case 2: Vector array operation

```
IppStatus, ippmExtract_va_32f(const Ipp32f* pSrc, int srcStride0,
   int srcStride2, Ipp32f* pDst, int len, int count);

IppStatus ippmExtract_va_64f(const Ipp64f* pSrc, int srcStride0,
   int srcStride2, Ipp64f* pDst, int len, int count);

IppStatus ippmExtract_va_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp32f* pDst, int len, int count);

IppStatus ippmExtract_va_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp64f* pDst, int len, int count);

IppStatus ippmExtract_va_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride2, Ipp32f* pDst, int len, int count);

IppStatus ippmExtract_va_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride2, Ipp64f* pDst, int len, int count);
```

Case 3: Matrix operation

```
IppStatus ippmExtract_m_32f(const Ipp32f* pSrc, int srcStride1, int
   srcStride2, Ipp32f* pDst, int width, int height);

IppStatus ippmExtract_m_64f(const Ipp64f* pSrc, int srcStride1, int
   srcStride2, Ipp64f* pDst, int width, int height);

IppStatus ippmExtract_m_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
   Ipp32f* pDst, int width, int height);

IppStatus ippmExtract_m_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
   Ipp64f* pDst, int width, int height);
```

Case 4: Transposed matrix operation

```
IppStatus ippmExtract_t_32f(const Ipp32f* pSrc, int srcStride1,
   int srcStride2, Ipp32f* pDst, int width, int height);

IppStatus ippmExtract_t_64f(const Ipp64f* pSrc, int srcStride1,
   int srcStride2, Ipp64f* pDst, int width, int height);

IppStatus ippmExtract_t_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
   Ipp32f* pDst, int width, int height);

IppStatus ippmExtract_t_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
   Ipp64f* pDst, int width, int height);
```

Case 5: Matrix array operation

```
IppStatus ippmExtract_ma_32f(const Ipp32f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp32f* pDst, int width, int height,
   int count);

IppStatus ippmExtract_ma_64f(const Ipp64f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp64f* pDst, int width, int height,
   int count);

IppStatus ippmExtract_ma_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp32f* pDst, int width, int height, int count);
```

```
IppStatus ippmExtract_ma_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp64f* pDst, int width, int height, int count);
IppStatus ippmExtract_ma_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride1, int srcStride2, Ipp32f* pDst, int width, int height,
   int count);
IppStatus ippmExtract_ma_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride1, int srcStride2, Ipp64f* pDst, int width, int height,
   int count);
```

Case 6: Transposed matrix array operation

```
IppStatus ippmExtract_ta_32f(const Ipp32f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp32f* pDst, int width, int height,
   int count);
IppStatus ippmExtract_ta_64f(const Ipp64f* pSrc, int srcStride0,
   int srcStride1, int srcStride2, Ipp64f* pDst, int width, int height,
   int count);
IppStatus ippmExtract_ta_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp32f* pDst, int width, int height, int count);
IppStatus ippmExtract_ta_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride0, Ipp64f* pDst, int width, int height, int count);
IppStatus ippmExtract_ta_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride1, int srcStride2, Ipp32f* pDst, int width, int height,
   int count);
IppStatus ippmExtract_ta_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride1, int srcStride2, Ipp64f* pDst, int width, int height,
   int count);
```

LoadIdentity

Initializes identity matrix.

Case 1: Matrix array operation

```
IppStatus ippmLoadIdentity_ma_32f(const Ipp32f* pDst, int dstStride0,
   int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmLoadIdentity_ma_64f(const Ipp64f* pDst, int dstStride0,
   int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmLoadIdentity_ma_32f_P(const Ipp32f** ppDst, int
   dstRoiShift, int dstStride2, int width, int height, int count);
IppStatus ippmLoadIdentity_ma_64f_P(const Ipp64f** ppDst, int
   dstRoiShift, int dstStride2, int width, int height, int count);
IppStatus ippmLoadIdentity_ma_32f_L(Ipp32f** ppDst, int dstRoiShift,
   int dstStride1, int dstStride2, int width, int height, int count);
IppStatus ippmLoadIdentity_ma_64f_L(Ipp64f** ppDst, int dstRoiShift,
   int dstStride1, int dstStride2, int width, int height, int count);
```

Vector Algebra Functions

Saxpy

Performs the “saxpy” operation on vectors.

Case 1: Vector - vector operation

```
IppStatus ippmSaxpy_vv_32f(const Ipp32f* pSrc1, int src1Stride2, Ipp32f
    scale, const Ipp32f* pSrc2, int src2Stride2, Ipp32f* pDst, int
    dstStride2, int len);

IppStatus ippmSaxpy_vv_64f(const Ipp64f* pSrc1, int src1Stride2, Ipp64f
    scale, const Ipp64f* pSrc2, int src2Stride2, Ipp64f* pDst, int
    dstStride2, int len);

IppStatus ippmSaxpy_vv_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    Ipp32f scale, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int len);

IppStatus ippmSaxpy_vv_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    Ipp64f scale, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int len);
```

Case 2: Vector - vector array operation

```
IppStatus ippmSaxpy_vva_32f(const Ipp32f* pSrc1, int src1Stride2, Ipp32f
    scale, const Ipp32f* pSrc2, int src2Stride0, int src2Stride2, Ipp32f*
    pDst, int dstStride0, int dstStride2, int len, int count);

IppStatus ippmSaxpy_vva_64f(const Ipp64f* pSrc1, int src1Stride2, Ipp64f
    scale, const Ipp64f* pSrc2, int src2Stride0, int src2Stride2, Ipp64f*
    pDst, int dstStride0, int dstStride2, int len, int count);

IppStatus ippmSaxpy_vva_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    Ipp32f scale, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmSaxpy_vva_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    Ipp64f scale, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmSaxpy_vva_32f_L(const Ipp32f* pSrc1, int src1Stride2,
    Ipp32f scale, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride2, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);

IppStatus ippmSaxpy_vva_64f_L(const Ipp64f* pSrc1, int src1Stride2,
    Ipp64f scale, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride2, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);
```

Case 3: Vector array - vector operation

```
IppStatus ippmSaxpy_vav_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride2, Ipp32f scale, const Ipp32f* pSrc2, int src2Stride2,
    Ipp32f* pDst, int dstStride0, int dstStride2, int len, int count);

IppStatus ippmSaxpy_vav_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride2, Ipp64f scale, const Ipp64f* pSrc2, int src2Stride2,
    Ipp64f* pDst, int dstStride0, int dstStride2, int len, int count);

IppStatus ippmSaxpy_vav_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, Ipp32f scale, const Ipp32f** ppSrc2, int
    src2RoiShift, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmSaxpy_vav_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, Ipp64f scale, const Ipp64f** ppSrc2, int
    src2RoiShift, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmSaxpy_vav_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride2, Ipp32f scale, const Ipp32f* pSrc2, int src2Stride2,
    Ipp32f** ppDst, int dstRoiShift, int dstStride2, int len, int count);

IppStatus ippmSaxpy_vav_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride2, Ipp64f scale, const Ipp64f* pSrc2, int src2Stride2,
    Ipp64f** ppDst, int dstRoiShift, int dstStride2, int len, int count);
```

Case 4: Vector array - vector array operation

```
IppStatus ippmSaxpy_vava_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride2, Ipp32f scale, const Ipp32f* pSrc2, int src2Stride0,
    int src2Stride2, Ipp32f* pDst, int dstStride0, int dstStride2, int
    len, int count);

IppStatus ippmSaxpy_vava_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride2, Ipp64f scale, const Ipp64f* pSrc2, int src2Stride0,
    int src2Stride2, Ipp64f* pDst, int dstStride0, int dstStride2, int
    len, int count);

IppStatus ippmSaxpy_vava_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, Ipp32f scale, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, Ipp32f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);

IppStatus ippmSaxpy_vava_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, Ipp64f scale, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, Ipp64f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);

IppStatus ippmSaxpy_vava_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride2, Ipp32f scale, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride2, int len, int count);
```

```
IppStatus ippmSaxpy_vava_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride2, Ipp64f scale, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride2, int len, int count);
```

Add

Adds constant to vector or vector to another vector.

Case 1: Vector - constant operation

```
IppStatus ippmAdd_vc_32f(const Ipp32f* pSrc, int srcStride2, Ipp32f val,
    Ipp32f* pDst, int dstStride2, int len);
IppStatus ippmAdd_vc_64f(const Ipp64f* pSrc, int srcStride2, Ipp64f val,
    Ipp64f* pDst, int dstStride2, int len);
IppStatus ippmAdd_vc_32f_P(const Ipp32f** ppSrc, int srcRoiShift, Ipp32f
    val, Ipp32f** ppDst, int dstRoiShift, int len);
IppStatus ippmAdd_vc_64f_P(const Ipp64f** ppSrc, int srcRoiShift, Ipp64f
    val, Ipp64f** ppDst, int dstRoiShift, int len);
```

Case 2: Vector array - constant operation

```
IppStatus ippmAdd_vac_32f(const Ipp32f* pSrc, int srcStride0, int
    srcStride2, Ipp32f val, Ipp32f* pDst, int dstStride0, int dstStride2,
    int len, int count);
IppStatus ippmAdd_vac_64f(const Ipp64f* pSrc, int srcStride0, int
    srcStride2, Ipp64f val, Ipp64f* pDst, int dstStride0, int dstStride2,
    int len, int count);
IppStatus ippmAdd_vac_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f val, Ipp32f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);
IppStatus ippmAdd_vac_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f val, Ipp64f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);
IppStatus ippmAdd_vac_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp32f val, Ipp32f** ppDst, int dstRoiShift, int
    dstStride2, int len, int count);
IppStatus ippmAdd_vac_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp64f val, Ipp64f** ppDst, int dstRoiShift, int
    dstStride2, int len, int count);
```

Case 3: Vector array - vector operation

```
IppStatus ippmAdd_vv_32f(const Ipp32f* pSrc1, int src1Stride2,
    const Ipp32f* pSrc2, int src2Stride2, Ipp32f* pDst, int dstStride2,
    int len);
IppStatus ippmAdd_vv_64f(const Ipp64f* pSrc1, int src1Stride2,
    const Ipp64f* pSrc2, int src2Stride2, Ipp64f* pDst, int dstStride2,
    int len);
```

```
IppStatus ippmAdd_vv_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f** ppDst, int
    dstRoiShift, int len);

IppStatus ippmAdd_vv_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f** ppDst, int
    dstRoiShift, int len);
```

Case 3: Vector array - vector operation

```
IppStatus ippmAdd_vav_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride2, Ipp32f* pDst,
    int dstStride0, int dstStride2, int len, int count);

IppStatus ippmAdd_vav_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride2, Ipp64f* pDst,
    int dstStride0, int dstStride2, int len, int count);

IppStatus ippmAdd_vav_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int len, int count);

IppStatus ippmAdd_vav_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int len, int count);

IppStatus ippmAdd_vav_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride2, Ipp32f**
    ppDst, int dstRoiShift, int dstStride2, int len, int count);

IppStatus ippmAdd_vav_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride2, Ipp64f**
    ppDst, int dstRoiShift, int dstStride2, int len, int count);
```

Case 4: Vector array - vector array operation

```
IppStatus ippmAdd_vava_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride0, int
    src2Stride2, Ipp32f* pDst, int dstStride0, int dstStride2, int len,
    int count);

IppStatus ippmAdd_vava_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride0, int
    src2Stride2, Ipp64f* pDst, int dstStride0, int dstStride2, int len,
    int count);

IppStatus ippmAdd_vava_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmAdd_vava_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);
```

```
IppStatus ippmAdd_vava_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride2, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride2, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);
IppStatus ippmAdd_vava_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride2, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride2, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);
```

Sub

Subtracts constant from vector, vector from constant, or vector from another vector.

Case 1: Vector - constant operation

```
IppStatus ippmSub_vc_32f(const Ipp32f* pSrc, int srcStride2, Ipp32f val,
    Ipp32f* pDst, int dstStride2, int len);
IppStatus ippmSub_vc_64f(const Ipp64f* pSrc, int srcStride2, Ipp64f val,
    Ipp64f* pDst, int dstStride2, int len);
IppStatus ippmSub_vc_32f_P(const Ipp32f** ppSrc, int srcRoiShift, Ipp32f
    val, Ipp32f** ppDst, int dstRoiShift, int len);
IppStatus ippmSub_vc_64f_P(const Ipp64f** ppSrc, int srcRoiShift, Ipp64f
    val, Ipp64f** ppDst, int dstRoiShift, int len);
```

Case 2: Vector array - constant operation

```
IppStatus ippmSub_vac_32f(const Ipp32f* pSrc, int srcStride0, int
    srcStride2, Ipp32f val, Ipp32f* pDst, int dstStride0, int dstStride2,
    int len, int count);
IppStatus ippmSub_vac_64f(const Ipp64f* pSrc, int srcStride0, int
    srcStride2, Ipp64f val, Ipp64f* pDst, int dstStride0, int dstStride2,
    int len, int count);
IppStatus ippmSub_vac_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f val, Ipp32f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);
IppStatus ippmSub_vac_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f val, Ipp64f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);
IppStatus ippmSub_vac_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp32f val, Ipp32f** ppDst, int dstRoiShift, int
    dstStride2, int len, int count);
IppStatus ippmSub_vac_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp64f val, Ipp64f** ppDst, int dstRoiShift, int
    dstStride2, int len, int count);
```

Case 3: Constant - vector operation

```
IppStatus ippmSub_cv_32f(const Ipp32f* pSrc, int srcStride2, Ipp32f val,
    Ipp32f* pDst, int dstStride2, int len);
```

```
IppStatus ippmSub_cv_64f(const Ipp64f* pSrc, int srcStride2, Ipp64f val,
    Ipp64f* pDst, int dstStride2, int len);

IppStatus ippmSub_cv_32f_P(const Ipp32f** ppSrc, int srcRoiShift, Ipp32f
    val, Ipp32f** ppDst, int dstRoiShift, int len);

IppStatus ippmSub_cv_64f_P(const Ipp64f** ppSrc, int srcRoiShift, Ipp64f
    val, Ipp64f** ppDst, int dstRoiShift, int len);
```

Case 4: Constant - vector array operation

```
IppStatus ippmSub_cva_32f(const Ipp32f* pSrc, int srcStride0, int
    srcStride2, Ipp32f val, Ipp32f* pDst, int dstStride0, int dstStride2,
    int len, int count);

IppStatus ippmSub_cva_64f(const Ipp64f* pSrc, int srcStride0, int
    srcStride2, Ipp64f val, Ipp64f* pDst, int dstStride0, int dstStride2,
    int len, int count);

IppStatus ippmSub_cva_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f val, Ipp32f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);

IppStatus ippmSub_cva_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f val, Ipp64f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);

IppStatus ippmSub_cva_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp32f val, Ipp32f** ppDst, int dstRoiShift, int
    dstStride2, int len, int count);

IppStatus ippmSub_cva_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp64f val, Ipp64f** ppDst, int dstRoiShift, int
    dstStride2, int len, int count);
```

Case 5: Vector - vector operation

```
IppStatus ippmSub_vv_32f(const Ipp32f* pSrc1, int src1Stride2,
    const Ipp32f* pSrc2, int src2Stride2, Ipp32f* pDst, int dstStride2,
    int len);

IppStatus ippmSub_vv_64f(const Ipp64f* pSrc1, int src1Stride2,
    const Ipp64f* pSrc2, int src2Stride2, Ipp64f* pDst, int dstStride2,
    int len);

IppStatus ippmSub_vv_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f** ppDst, int
    dstRoiShift, int len);

IppStatus ippmSub_vv_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f** ppDst, int
    dstRoiShift, int len);
```

Case 6: Vector - vector array operation

```
IppStatus ippmSub_vva_32f(const Ipp32f* pSrc1, int src1Stride2,
    const Ipp32f* pSrc2, int src2Stride0, int src2Stride2, Ipp32f* pDst,
    int dstStride0, int dstStride2, int len, int count);
```

```
IppStatus ippmSub_vva_64f(const Ipp64f* pSrc1, int src1Stride2,
   const Ipp64f* pSrc2, int src2Stride0, int src2Stride2, Ipp64f* pDst,
   int dstStride0, int dstStride2, int len, int count);

IppStatus ippmSub_vva_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
   const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp32f**
   ppDst, int dstRoiShift, int dstStride0, int len, int count);

IppStatus ippmSub_vva_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
   const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp64f**
   ppDst, int dstRoiShift, int dstStride0, int len, int count);

IppStatus ippmSub_vva_32f_L(const Ipp32f* pSrc1, int src1Stride2,
   const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride2, Ipp32f**
   ppDst, int dstRoiShift, int dstStride2, int len, int count);

IppStatus ippmSub_vva_64f_L(const Ipp64f* pSrc1, int src1Stride2,
   const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride2, Ipp64f**
   ppDst, int dstRoiShift, int dstStride2, int len, int count);
```

Case 7: Vector array - vector operation

```
IppStatus ippmSub_vav_32f(const Ipp32f* pSrc1, int src1Stride0,
   int src1Stride2, const Ipp32f* pSrc2, int src2Stride2, Ipp32f* pDst,
   int dstStride0, int dstStride2, int len, int count);

IppStatus ippmSub_vav_64f(const Ipp64f* pSrc1, int src1Stride0,
   int src1Stride2, const Ipp64f* pSrc2, int src2Stride2, Ipp64f* pDst,
   int dstStride0, int dstStride2, int len, int count);

IppStatus ippmSub_vav_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
   int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
   ppDst, int dstRoiShift, int dstStride0, int len, int count);

IppStatus ippmSub_vav_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
   int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
   ppDst, int dstRoiShift, int dstStride0, int len, int count);

IppStatus ippmSub_vav_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
   int src1Stride2, const Ipp32f* pSrc2, int src2Stride2, Ipp32f**
   ppDst, int dstRoiShift, int dstStride2, int len, int count);

IppStatus ippmSub_vav_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
   int src1Stride2, const Ipp64f* pSrc2, int src2Stride2, Ipp64f**
   ppDst, int dstRoiShift, int dstStride2, int len, int count);
```

Case 8: Vector array - vector array operation

```
IppStatus ippmSub_vava_32f(const Ipp32f* pSrc1, int src1Stride0,
   int src1Stride2, const Ipp32f* pSrc2, int src2Stride0, int
   src2Stride2, Ipp32f* pDst, int dstStride0, int dstStride2, int len,
   int count);

IppStatus ippmSub_vava_64f(const Ipp64f* pSrc1, int src1Stride0,
   int src1Stride2, const Ipp64f* pSrc2, int src2Stride0, int
   src2Stride2, Ipp64f* pDst, int dstStride0, int dstStride2, int len,
   int count);
```

```
IppStatus ippmSub_vava_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmSub_vava_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    len, int count);

IppStatus ippmSub_vava_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride2, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride2, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);

IppStatus ippmSub_vava_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride2, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride2, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);
```

Mul

Multiplies vector by constant.

Case 1: Vector - constant operation

```
IppStatus ippmMul_vc_32f(const Ipp32f* pSrc, int srcStride2, Ipp32f val,
    Ipp32f* pDst, int dstStride2, int len);

IppStatus ippmMul_vc_64f(const Ipp64f* pSrc, int srcStride2, Ipp64f val,
    Ipp64f* pDst, int dstStride2, int len);

IppStatus ippmMul_vc_32f_P(const Ipp32f** ppSrc, int srcRoiShift, Ipp32f
    val, Ipp32f** ppDst, int dstRoiShift, int len);

IppStatus ippmMul_vc_64f_P(const Ipp64f** ppSrc, int srcRoiShift, Ipp64f
    val, Ipp64f** ppDst, int dstRoiShift, int len);
```

Case 2: Vector array - constant operation

```
IppStatus ippmMul_vac_32f(const Ipp32f* pSrc, int srcStride0, int
    srcStride2, Ipp32f val, Ipp32f* pDst, int dstStride0, int dstStride2,
    int len, int count);

IppStatus ippmMul_vac_64f(const Ipp64f* pSrc, int srcStride0, int
    srcStride2, Ipp64f val, Ipp64f* pDst, int dstStride0, int dstStride2,
    int len, int count);

IppStatus ippmMul_vac_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f val, Ipp32f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);

IppStatus ippmMul_vac_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f val, Ipp64f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);

IppStatus ippmMul_vac_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp32f val, Ipp32f** ppDst, int dstRoiShift, int
    dstStride2, int len, int count);
```

```
IppStatus ippmMul_vac_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride2, Ipp64f val, Ipp64f** ppDst, int dstRoiShift, int
   dstStride2, int len, int count);
```

CrossProduct

Computes cross product of two 3D vectors.

Case 1: Vector - vector operation

```
IppStatus ippmCrossProduct_vv_32f(const Ipp32f* pSrc1, int src1Stride2,
   const Ipp32f* pSrc2, int src2Stride2, Ipp32f* pDst, int dstStride2);

IppStatus ippmCrossProduct_vv_64f(const Ipp64f* pSrc1, int src1Stride2,
   const Ipp64f* pSrc2, int src2Stride2, Ipp64f* pDst, int dstStride2);

IppStatus ippmCrossProduct_vv_32f_P(const Ipp32f** ppSrc1, int
   src1RoiShift, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
   ppDst, int dstRoiShift);

IppStatus ippmCrossProduct_vv_64f_P(const Ipp64f** ppSrc1, int
   src1RoiShift, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
   ppDst, int dstRoiShift);
```

Case 2: Vector - vector array operation

```
IppStatus ippmCrossProduct_vva_32f(const Ipp32f* pSrc1, int src1Stride2,
   const Ipp32f* pSrc2, int src2Stride0, int src2Stride2, Ipp32f* pDst,
   int dstStride0, int dstStride2, int count);

IppStatus ippmCrossProduct_vva_64f(const Ipp64f* pSrc1, int src1Stride2,
   const Ipp64f* pSrc2, int src2Stride0, int src2Stride2, Ipp64f* pDst,
   int dstStride0, int dstStride2, int count);

IppStatus ippmCrossProduct_vva_32f_P(const Ipp32f** ppSrc1, int
   src1RoiShift, const Ipp32f** ppSrc2, int src2RoiShift, int
   src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
   count);

IppStatus ippmCrossProduct_vva_64f_P(const Ipp64f** ppSrc1, int
   src1RoiShift, const Ipp64f** ppSrc2, int src2RoiShift, int
   src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
   count);

IppStatus ippmCrossProduct_vva_32f_L(const Ipp32f* pSrc1, int
   src1Stride2, const Ipp32f** ppSrc2, int src2RoiShift, int
   src2Stride2, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int
   count);

IppStatus ippmCrossProduct_vva_64f_L(const Ipp64f* pSrc1, int
   src1Stride2, const Ipp64f** ppSrc2, int src2RoiShift, int
   src2Stride2, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int
   count);
```

Case 3: Vector array - vector operation

```
IppStatus ippmCrossProduct_vav_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride2, Ipp32f* pDst,
    int dstStride0, int dstStride2, int count);

IppStatus ippmCrossProduct_vav_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride2, Ipp64f* pDst,
    int dstStride0, int dstStride2, int count);

IppStatus ippmCrossProduct_vav_32f_P(const Ipp32f** ppSrc1, int
    src1RoiShift, int src1Stride0, const Ipp32f** ppSrc2, int
    src2RoiShift, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    count);

IppStatus ippmCrossProduct_vav_64f_P(const Ipp64f** ppSrc1, int
    src1RoiShift, int src1Stride0, const Ipp64f** ppSrc2, int
    src2RoiShift, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    count);

IppStatus ippmCrossProduct_vav_32f_L(const Ipp32f** ppSrc1, int
    src1RoiShift, int src1Stride2, const Ipp32f* pSrc2, int src2Stride2,
    Ipp32f** ppDst, int dstRoiShift, int dstStride2, int count);

IppStatus ippmCrossProduct_vav_64f_L(const Ipp64f** ppSrc1, int
    src1RoiShift, int src1Stride2, const Ipp64f* pSrc2, int src2Stride2,
    Ipp64f** ppDst, int dstRoiShift, int dstStride2, int count);
```

Case 4: Vector array - vector array operation

```
IppStatus ippmCrossProduct_vava_32f(const Ipp32f* pSrc1, int
    src1Stride0, int src1Stride2, const Ipp32f* pSrc2, int src2Stride0,
    int src2Stride2, Ipp32f* pDst, int dstStride0, int dstStride2, int
    count);

IppStatus ippmCrossProduct_vava_64f(const Ipp64f* pSrc1, int
    src1Stride0, int src1Stride2, const Ipp64f* pSrc2, int src2Stride0,
    int src2Stride2, Ipp64f* pDst, int dstStride0, int dstStride2, int
    count);

IppStatus ippmCrossProduct_vava_32f_P(const Ipp32f** ppSrc1, int
    src1RoiShift, int src1Stride0, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, Ipp32f** pDst, int dstRoiShift, int
    dstStride0, int count);

IppStatus ippmCrossProduct_vava_64f_P(const Ipp64f** ppSrc1, int
    src1RoiShift, int src1Stride0, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, Ipp64f** ppDst, int dstRoiShift, int
    dstStride0, int count);

IppStatus ippmCrossProduct_vava_32f_L(const Ipp32f** ppSrc1, int
    src1RoiShift, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride2, int count);
```

```
IppStatus ippmCrossProduct_vava_64f_L(const Ipp64f** ppSrc1, int
   src1RoiShift, int src1Stride2, const Ipp64f** ppSrc2, int
   src2RoiShift, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
   dstStride2, int count);
```

DotProduct

Computes dot product of two vectors.

Case 1: Vector - vector operation

```
IppStatus ippmDotProduct_vv_32f(const Ipp32f* pSrc1, int src1Stride2,
   const Ipp32f* pSrc2, int src2Stride2, Ipp32f* pDst, int len);

IppStatus ippmDotProduct_vv_64f(const Ipp64f* pSrc1, int src1Stride2,
   const Ipp64f* pSrc2, int src2Stride2, Ipp64f* pDst, int len);

IppStatus ippmDotProduct_vv_32f_P(const Ipp32f** ppSrc1, int
   src1RoiShift, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f* pDst,
   int len);

IppStatus ippmDotProduct_vv_64f_P(const Ipp64f** ppSrc1, int
   src1RoiShift, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f* pDst,
   int len);
```

Case 2: Vector array - vector operation

```
IppStatus ippmDotProduct_vav_32f(const Ipp32f* pSrc1, int src1Stride0,
   int src1Stride2, const Ipp32f* pSrc2, int src2Stride2, Ipp32f* pDst,
   int len, int count);

IppStatus ippmDotProduct_vav_64f(const Ipp64f* pSrc1, int src1Stride0,
   int src1Stride2, const Ipp64f* pSrc2, int src2Stride2, Ipp64f* pDst,
   int len, int count);

IppStatus ippmDotProduct_vav_32f_P(const Ipp32f** ppSrc1, int
   src1RoiShift, int src1Stride0, const Ipp32f** ppSrc2, int
   src2RoiShift, Ipp32f* pDst, int len, int count);

IppStatus ippmDotProduct_vav_64f_P(const Ipp64f** ppSrc1, int
   src1RoiShift, int src1Stride0, const Ipp64f** ppSrc2, int
   src2RoiShift, Ipp64f* pDst, int len, int count);

IppStatus ippmDotProduct_vav_32f_L(const Ipp32f** ppSrc1, int
   src1RoiShift, int src1Stride2, const Ipp32f* pSrc2, int src2Stride2,
   Ipp32f* pDst, int len, int count);

IppStatus ippmDotProduct_vav_64f_L(const Ipp64f** ppSrc1, int
   src1RoiShift, int src1Stride2, const Ipp64f* pSrc2, int src2Stride2,
   Ipp64f* pDst, int len, int count);
```

Case 3: Vector array - vector array operation

```
IppStatus ippmDotProduct_vava_32f(const Ipp32f* pSrc1, int src1Stride0,
   int src1Stride2, const Ipp32f* pSrc2, int src2Stride0, int
   src2Stride2, Ipp32f* pDst, int len, int count);
```

```
IppStatus ippmDotProduct_vava_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride0, int
    src2Stride2, Ipp64f* pDst, int len, int count);
IppStatus ippmDotProduct_vava_32f_P(const Ipp32f** ppSrc1, int
    src1RoiShift, int src1Stride0, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, Ipp32f* pDst, int len, int count);
IppStatus ippmDotProduct_vava_64f_P(const Ipp64f** ppSrc1, int
    src1RoiShift, int src1Stride0, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, Ipp64f* pDst, int len, int count);
IppStatus ippmDotProduct_vava_32f_L(const Ipp32f** ppSrc1, int
    src1RoiShift, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride2, Ipp32f* pDst, int len, int count);
IppStatus ippmDotProduct_vava_64f_L(const Ipp64f** ppSrc1, int
    src1RoiShift, int src1Stride2, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride2, Ipp64f* pDst, int len, int count);
```

L2Norm

Computes vector L2 norm.

Case 1: Vector operation

```
IppStatus ippmL2Norm_v_32f(const Ipp32f* pSrc, int srcStride2, Ipp32f*
    pDst, int len);
IppStatus ippmL2Norm_v_64f(const Ipp64f* pSrs, int srcStride2, Ipp64f*
    pDst, int len);
IppStatus ippmL2Norm_v_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    Ipp32f* pDst, int len);
IppStatus ippmL2Norm_v_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    Ipp64f* pDst, int len);
```

Case 2: Vector array operation

```
IppStatus ippmL2Norm_va_32f(const Ipp32f* pSrc, int srcStride0, int
    srcStride2, Ipp32f* pDst, int len, int count);
IppStatus ippmL2Norm_va_64f(const Ipp64f* pSrc, int srcStride0, int
    srcStride2, Ipp64f* pDst, int len, int count);
IppStatus ippmL2Norm_va_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f* pDst, int len, int count);
IppStatus ippmL2Norm_va_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f* pDst, int len, int count);
IppStatus ippmL2Norm_va_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp32f* pDst, int len, int count);
IppStatus ippmL2Norm_va_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride2, Ipp64f* pDst, int len, int count);
```

LComb

Composes linear combination of two vectors.

Case 1: Vector - vector operation

```
IppStatus ippmLComb_vv_32f(const Ipp32f* pSrc1, int src1Stride2, Ipp32f
    scale1, const Ipp32f* pSrc2, int src2Stride2, Ipp32f scale2, Ipp32f*
    pDst, int dstStride2, int len);

IppStatus ippmLComb_vv_64f(const Ipp64f* pSrc1, int src1Stride2, Ipp64f
    scale1, const Ipp64f* pSrc2, int src2Stride2, Ipp64f scale2, Ipp64f*
    pDst, int dstStride2, int len);

IppStatus ippmLComb_vv_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    Ipp32f scale1, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f
    scale2, Ipp32f** ppDst, int dstRoiShift, int len);

IppStatus ippmLComb_vv_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    Ipp64f scale1, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f
    scale2, Ipp64f** ppDst, int dstRoiShift, int len);
```

Case 2: Vector array - vector operation

```
IppStatus ippmLComb_vav_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride2, Ipp32f scale1, const Ipp32f* pSrc2, int src2Stride2,
    Ipp32f scale2, Ipp32f* pDst, int dstStride0, int dstStride2, int len,
    int count);

IppStatus ippmLComb_vav_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride2, Ipp64f scale1, const Ipp64f* pSrc2, int src2Stride2,
    Ipp64f scale2, Ipp64f* pDst, int dstStride0, int dstStride2, int len,
    int count);

IppStatus ippmLComb_vav_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, Ipp32f scale1, const Ipp32f** ppSrc2, int
    src2RoiShift, Ipp32f scale2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);

IppStatus ippmLComb_vav_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, Ipp64f scale1, const Ipp64f** ppSrc2, int
    src2RoiShift, Ipp64f scale2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride0, int len, int count);

IppStatus ippmLComb_vav_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride2, Ipp32f scale1, const Ipp32f* pSrc2, int src2Stride2,
    Ipp32f scale2, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);

IppStatus ippmLComb_vav_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride2, Ipp64f scale1, const Ipp64f* pSrc2, int src2Stride2,
    Ipp64f scale2, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int
    len, int count);
```

Case 3: Vector array - vector array operation

```
IppStatus ippmLComb_vava_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride2, Ipp32f scale1, const Ipp32f* pSrc2, int src2Stride0,
    int src2Stride2, Ipp32f scale2, Ipp32f* pDst, int dstStride0,
    int dstStride2, int len, int count);

IppStatus ippmLComb_vava_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride2, Ipp64f scale1, const Ipp64f* pSrc2, int src2Stride0,
    int src2Stride2, Ipp64f scale2, Ipp64f* pDst, int dstStride0,
    int dstStride2, int len, int count);

IppStatus ippmLComb_vava_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, Ipp32f scale1, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, Ipp32f scale2, Ipp32f** ppDst, int
    dstRoiShift,  int dstStride0, int len, int count);

IppStatus ippmLComb_vava_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, Ipp64f scale1, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, Ipp64f scale2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride0, int len, int count);

IppStatus ippmLComb_vava_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride2, Ipp32f scale1, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride2, Ipp32f scale2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride2, int len, int count);

IppStatus ippmLComb_vava_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride2, Ipp64f scale1, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride2, Ipp64f scale2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride2, int len, int count);
```

Matrix Algebra Functions

Transpose

Performs matrix transposition.

Case 1: Matrix operation

```
IppStatus ippmTranspose_m_32f(const Ipp32f* pSrc, int srcStride1,
    int srcStride2, int width, int height, Ipp32f* pDst, int dstStride1,
    int dstStride2);

IppStatus ippmTranspose_m_64f(const Ipp64f* pSrc, int srcStride1,
    int srcStride2, int width, int height, Ipp64f* pDst, int dstStride1,
    int dstStride2);

IppStatus ippmTranspose_m_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int width, int height, Ipp32f** ppDst, int dstRoiShift);

IppStatus ippmTranspose_m_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int width, int height, Ipp64f** ppDst, int dstRoiShift);
```

Case 2: Matrix array operation

```
IppStatus ippmTranspose_ma_32f(const Ipp32f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, int width, int height, Ipp32f* pDst,
    int dstStride0, int dstStride1, int dstStride2, int count);
IppStatus ippmTranspose_ma_64f(const Ipp64f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, int width, int height, Ipp64f* pDst,
    int dstStride0, int dstStride1, int dstStride2, int count);
IppStatus ippmTranspose_ma_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, int width, int height, Ipp32f** ppDst, int
    dstRoiShift, int dstStride0, int count);
IppStatus ippmTranspose_ma_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, int width, int height, Ipp64f** ppDst, int
    dstRoiShift, int dstStride0, int count);
IppStatus ippmTranspose_ma_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, int width, int height, Ipp32f**
    ppDst, int dstRoiShift, int dstStride1, int dstStride2, int count);
IppStatus ippmTranspose_ma_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, int width, int height, Ipp64f**
    ppDst, int dstRoiShift, int dstStride1, int dstStride2, int count);
```

Invert

Computes matrix inverse.

Case 1: Matrix operation

```
IppStatus ippmInvert_m_32f(const Ipp32f* pSrc, int srcStride1, int
    srcStride2, Ipp32f* pBuffer, Ipp32f* pDst, int dstStride1, int
    dstStride2, int widthHeight);
IppStatus ippmInvert_m_64f(const Ipp64f* pSrc, int srcStride1, int
    srcStride2, Ipp64f* pBuffer, Ipp64f* pDst, int dstStride1, int
    dstStride2, int widthHeight);
IppStatus ippmInvert_m_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    Ipp32f* pBuffer, Ipp32f** ppDst, int dstRoiShift, int widthHeight);
IppStatus ippmInvert_m_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    Ipp64f* pBuffer, Ipp64f** ppDst, int dstRoiShift, int widthHeight);
```

Case 2: Matrix array operation

```
IppStatus ippmInvert_ma_32f(const Ipp32f* pSrc, int srcStride0, int
    srcStride1, int srcStride2, Ipp32f* pBuffer, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int widthHeight, int
    count);
IppStatus ippmInvert_ma_64f(const Ipp64f* pSrc, int srcStride0, int
    srcStride1, int srcStride2, Ipp64f* pBuffer, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int widthHeight, int
    count);
```

```
IppStatus ippmInvert_ma_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f* pBuffer, Ipp32f** ppDst, int dstRoiShift,
    int dstStride0, int widthHeight, int count);
IppStatus ippmInvert_ma_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f* pBuffer, Ipp64f** ppDst, int dstRoiShift,
    int dstStride0, int widthHeight, int count);
IppStatus ippmInvert_ma_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, Ipp32f* pBuffer, Ipp32f** ppDst,
    int dstRoiShift, int dstStride1, int dstStride2, int widthHeight,
    int count);
IppStatus ippmInvert_ma_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, Ipp64f* pBuffer, Ipp64f** ppDst,
    int dstRoiShift, int dstStride1, int dstStride2, int widthHeight,
    int count);
```

FrobNorm

Computes matrix Frobenius.

Case 1: Matrix operation

```
IppStatus ippmFrobNorm_m_32f(const Ipp32f* pSrc, int srcStride1,
    int srcStride2, int width, int height, Ipp32f* pDst);
IppStatus ippmFrobNorm_m_64f(const Ipp64f* pSrc, int srcStride1,
    int srcStride2, int width, int height, Ipp64f* pDst);
IppStatus ippmFrobNorm_m_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int width, int height, Ipp32f* pDst);
IppStatus ippmFrobNorm_m_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int width, int height, Ipp64f* pDst);
```

Case 2: Matrix array operation

```
IppStatus ippmFrobNorm_ma_32f(const Ipp32f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, int width, int height, Ipp32f* pDst,
    int count);
IppStatus ippmFrobNorm_ma_64f(const Ipp64f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, int width, int height, Ipp64f* pDst,
    int count);
IppStatus ippmFrobNorm_ma_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, int width, int height, Ipp32f* pDst, int count);
IppStatus ippmFrobNorm_ma_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, int width, int height, Ipp64f* pDst, int count);
IppStatus ippmFrobNorm_ma_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, int width, int height, Ipp32f* pDst,
    int count);
IppStatus ippmFrobNorm_ma_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, int width, int height, Ipp64f* pDst,
    int count);
```

Det

Computes matrix determinant.

Case 1: Matrix operation

```
IppStatus ippmDet_m_32f(const Ipp32f* pSrc, int srcStride1, int
   srcStride2, int widthHeight, Ipp32f* pBuffer, Ipp32f* pDst);
IppStatus ippmDet_m_64f(const Ipp64f* pSrc, int srcStride1, int
   srcStride2, int widthHeight, Ipp64f* pBuffer, Ipp64f* pDst);
IppStatus ippmDet_m_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
   int widthHeight, Ipp32f* pBuffer, Ipp32f* pDst);
IppStatus ippmDet_m_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
   int widthHeight, Ipp64f* pBuffer, Ipp64f* pDst);
```

Case 2: Matrix array operation

```
IppStatus ippmDet_ma_32f(const Ipp32f* pSrc, int srcStride0, int
   srcStride1, int srcStride2, int widthHeight, Ipp32f* pBuffer, Ipp32f*
   pDst, int count);
IppStatus ippmDet_ma_64f(const Ipp64f* pSrc, int srcStride0, int
   srcStride1, int srcStride2, int widthHeight, Ipp64f* pBuffer, Ipp64f*
   pDst, int count);
IppStatus ippmDet_ma_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride0, int widthHeight, Ipp32f* pBuffer, Ipp32f* pDst, int
   count);
IppStatus ippmDet_ma_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride0, int widthHeight, Ipp64f* pBuffer, Ipp64f* pDst, int
   count);
IppStatus ippmDet_ma_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
   int srcStride1, int srcStride2, int widthHeight, Ipp32f* pBuffer,
   Ipp32f* pDst, int count);
IppStatus ippmDet_ma_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
   int srcStride1, int srcStride2, int widthHeight, Ipp64f* pBuffer,
   Ipp64f* pDst, int count);
```

Trace

Computes matrix trace.

Case 1: Matrix operation

```
IppStatus ippmTrace_m_32f(const Ipp32f* pSrc, int srcStride1, int
   srcStride2, int widthHeight, Ipp32f* pDst);
IppStatus ippmTrace_m_64f(const Ipp64f* pSrc, int srcStride1, int
   srcStride2, int widthHeight, Ipp64f* pDst);
IppStatus ippmTrace_m_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
   int widthHeight, Ipp32f* pDst);
```

```
IppStatus ippmTrace_m_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int widthHeight, Ipp64f* pDst);
```

Case 2: Matrix array operation

```
IppStatus ippmTrace_ma_32f(const Ipp32f* pSrc, int srcStride0, int
    srcStride1, int srcStride2, int widthHeight, Ipp32f* pDst, int
    count);

IppStatus ippmTrace_ma_64f(const Ipp64f* pSrc, int srcStride0, int
    srcStride1, int srcStride2, int widthHeight, Ipp64f* pDst, int
    count);

IppStatus ippmTrace_ma_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, int widthHeight, Ipp32f* pDst, int count);

IppStatus ippmTrace_ma_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, int widthHeight, Ipp64f* pDst, int count);

IppStatus ippmTrace_ma_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, int widthHeight, Ipp32f* pDst, int
    count);

IppStatus ippmTrace_ma_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, int widthHeight, Ipp64f* pDst, int
    count);
```

Mul

Multiplies matrix by a constant, a vector or another matrix.

Case 1: Matrix - constant operation

```
IppStatus ippmMul_mc_32f(const Ipp32f* pSrc, int srcStride1, int
    srcStride2, Ipp32f val, Ipp32f* pDst, int dstStride1, int dstStride2,
    int width, int height);

IppStatus ippmMul_mc_64f(const Ipp64f* pSrc, int srcStride1, int
    srcStride2, Ipp64f val, Ipp64f* pDst, int dstStride1, int dstStride2,
    int width, int height);

IppStatus ippmMul_mc_32f_P(const Ipp32f** ppSrc, int srcRoiShift, Ipp32f
    val, Ipp32f** ppDst, int dstRoiShift, int width, int height);

IppStatus ippmMul_mc_64f_P(const Ipp64f** ppSrc, int srcRoiShift, Ipp64f
    val, Ipp64f** ppDst, int dstRoiShift, int width, int height);
```

Case 2: Transposed matrix - constant operation

```
IppStatus ippmMul_tc_32f(const Ipp32f* pSrc, int srcStride1, int
    srcStride2, Ipp32f val, Ipp32f* pDst, int dstStride1, int dstStride2,
    int width, int height);

IppStatus ippmMul_tc_64f(const Ipp64f* pSrc, int srcStride1, int
    srcStride2, Ipp64f val, Ipp64f* pDst, int dstStride1, int dstStride2,
    int width, int height);

IppStatus ippmMul_tc_32f_P(const Ipp32f** ppSrc, int srcRoiShift, Ipp32f
    val, Ipp32f** ppDst, int dstRoiShift, int width, int height);
```

IppStatus ippmMul_tc_64f_P(const Ipp64f** *ppSrc*, int *srcRoiShift*, Ipp64f *val*, Ipp64f** *pDst*, int *dstRoiShift*, int *width*, int *height*);

Case 3: Matrix array - constant operation

IppStatus ippmMul_mac_32f(const Ipp32f* *pSrc*, int *srcStride0*, int *srcStride1*, int *srcStride2*, Ipp32f *val*, Ipp32f* *pDst*, int *dstStride0*, int *dstStride1*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmMul_mac_64f(const Ipp64f* *pSrc*, int *srcStride0*, int *srcStride1*, int *srcStride2*, Ipp64f *val*, Ipp64f* *pDst*, int *dstStride0*, int *dstStride1*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmMul_mac_32f_P(const Ipp32f** *ppSrc*, int *srcRoiShift*, int *srcStride0*, Ipp32f *val*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *width*, int *height*, int *count*);

IppStatus ippmMul_mac_64f_P(const Ipp64f** *ppSrc*, int *srcRoiShift*, int *srcStride0*, Ipp64f *val*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *width*, int *height*, int *count*);

IppStatus ippmMul_mac_32f_L(const Ipp32f** *ppSrc*, int *srcRoiShift*, int *srcStride1*, int *srcStride2*, Ipp32f *val*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride1*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmMul_mac_64f_L(const Ipp64f** *ppSrc*, int *srcRoiShift*, int *srcStride1*, int *srcStride2*, Ipp64f *val*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride1*, int *dstStride2*, int *width*, int *height*, int *count*);

Case 4: Transposed matrix array - constant operation

IppStatus ippmMul_tac_32f(const Ipp32f* *pSrc*, int *srcStride0*, int *srcStride1*, int *srcStride2*, Ipp32f *val*, Ipp32f* *pDst*, int *dstStride0*, int *dstStride1*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmMul_tac_64f(const Ipp64f* *pSrc*, int *srcStride0*, int *srcStride1*, int *srcStride2*, Ipp64f *val*, Ipp64f* *pDst*, int *dstStride0*, int *dstStride1*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmMul_tac_32f_P(const Ipp32f** *ppSrc*, int *srcRoiShift*, int *srcStride0*, Ipp32f *val*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *width*, int *height*, int *count*);

IppStatus ippmMul_tac_64f_P(const Ipp64f** *ppSrc*, int *srcRoiShift*, int *srcStride0*, Ipp64f *val*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *width*, int *height*, int *count*);

IppStatus ippmMul_tac_32f_L(const Ipp32f** *ppSrc*, int *srcRoiShift*, int *srcStride1*, int *srcStride2*, Ipp32f *val*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride1*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmMul_tac_64f_L(const Ipp64f** *ppSrc*, int *srcRoiShift*, int *srcStride1*, int *srcStride2*, Ipp64f *val*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride1*, int *dstStride2*, int *width*, int *height*, int *count*);

Case 5: Matrix - vector operation

```
IppStatus ippmMul_mv_32f(const Ipp32f* pSrc1, int src1Stride1, int
    src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2, int
    src2Stride2, int src2Len, Ipp32f* pDst, int dstStride2);

IppStatus ippmMul_mv_64f(const Ipp64f* pSrc1, int src1Stride1, int
    src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2, int
    src2Stride2, int src2Len, Ipp64f* pDst, int dstStride2);

IppStatus ippmMul_mv_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Len, Ipp32f** ppDst, int dstRoiShift);

IppStatus ippmMul_mv_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Len, Ipp64f** ppDst, int dstRoiShift);
```

Case 6: Transposed matrix - vector operation

```
IppStatus ippmMul_tv_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride2, int src2Len, Ipp32f* pDst, int dstStride2);

IppStatus ippmMul_tv_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride2, int src2Len, Ipp64f* pDst, int dstStride2);

IppStatus ippmMul_tv_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Len, Ipp32f** ppDst, int dstRoiShift);

IppStatus ippmMul_tv_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** pSrc2, int
    src2RoiShift, int src2Len, Ipp64f** ppDst, int dstRoiShift);
```

Case 7: Matrix - vector array operation

```
IppStatus ippmMul_mva_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride0, int src2Stride2, int src2Len, Ipp32f* pDst, int
    dstStride0, int dstStride2, int count);

IppStatus ippmMul_mva_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride0, int src2Stride2, int src2Len, Ipp64f* pDst, int
    dstStride0, int dstStride2, int count);

IppStatus ippmMul_mva_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Len, Ipp32f** ppDst, int
    dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mva_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Len, Ipp64f** ppDst, int
    dstRoiShift, int dstStride0, int count);
```

```
IppStatus ippmMul_mva_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride2, int src2Len, Ipp32f**
    ppDst, int dstRoiShift, int dstStride2, int count);

IppStatus ippmMul_mva_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride2, int src2Len, Ipp64f**
    ppDst, int dstRoiShift, int dstStride2, int count);
```

Case 8: Transposed matrix - vector array operation

```
IppStatus ippmMul_tva_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride0, int src2Stride2, int src2Len, Ipp32f* pDst, int
    dstStride0, int dstStride2, int count);

IppStatus ippmMul_tva_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride0, int src2Stride2, int src2Len, Ipp64f* pDst, int
    dstStride0, int dstStride2, int count);

IppStatus ippmMul_tva_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Len, Ipp32f** ppDst, int
    dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tva_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Len, Ipp64f** ppDst, int
    dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tva_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride2, int src2Len, Ipp32f**
    ppDst, int dstRoiShift, int dstStride2, int count);

IppStatus ippmMul_tva_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride2, int src2Len, Ipp64f**
    ppDst, int dstRoiShift, int dstStride2, int count);
```

Case 9: Matrix array - vector operation

```
IIppStatus ippmMul_mav_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride2, int src2Len, Ipp32f* pDst,
    int dstStride0, int dstStride2, int count);

IppStatus ippmMul_mav_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride2, int src2Len, Ipp64f* pDst,
    int dstStride0, int dstStride2, int count);
```

```
IppStatus ippmMul_mav_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Len, Ipp32f** ppDst, int
    dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mav_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Len, Ipp64f** ppDst, int
    dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mav_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride2, int src2Len, Ipp32f** ppDst,
    int dstRoiShift, int dstStride2, int count);

IppStatus ippmMul_mav_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride2, int src2Len, Ipp64f** ppDst,
    int dstRoiShift, int dstStride2, int count);
```

Case 10: Transposed matrix array - vector operation

```
IppStatus ippmMul_tav_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride2, int src2Len, Ipp32f* pDst,
    int dstStride0, int dstStride2, int count);

IppStatus ippmMul_tav_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride2, int src2Len, Ipp64f* pDst,
    int dstStride0, int dstStride2, int count);

IppStatus ippmMul_tav_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Len, Ipp32f** ppDst, int
    dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tav_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Len, Ipp64f** ppDst, int
    dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tav_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride2, int src2Len, Ipp32f** ppDst,
    int dstRoiShift, int dstStride2, int count);

IppStatus ippmMul_tav_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride2, int src2Len, Ipp64f** ppDst,
    int dstRoiShift, int dstStride2, int count);
```

Case 11: Matrix array - vector array operation

```
IppStatus ippmMul_mava_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride0, int src2Stride2, int src2Len,
    Ipp32f* pDst, int dstStride0, int dstStride2, int count);

IppStatus ippmMul_mava_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride0, int src2Stride2, int src2Len,
    Ipp64f* pDst, int dstStride0, int dstStride2, int count);

IppStatus ippmMul_mava_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride0, int src2Len, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mava_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride0, int src2Len, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mava_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride2, int
    src2Len, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int count);

IppStatus ippmMul_mava_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride2, int
    src2Len, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int count);
```

Case 12: Transposed matrix array - vector array operation

```
IppStatus ippmMul_tava_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride0, int src2Stride2, int src2Len,
    Ipp32f* pDst, int dstStride0, int dstStride2, int count);

IppStatus ippmMul_tava_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride0, int src2Stride2, int src2Len,
    Ipp64f* pDst, int dstStride0, int dstStride2, int count);

IppStatus ippmMul_tava_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride0, int src2Len, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tava_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride0, int src2Len, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int count);
```

```
IppStatus ippmMul_tava_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride2, int
    src2Len, Ipp32f** ppDst, int dstRoiShift, int dstStride2, int count);
IppStatus ippmMul_tava_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride2, int
    src2Len, Ipp64f** ppDst, int dstRoiShift, int dstStride2, int count);
```

Case 13: Matrix - matrix operation

```
IppStatus ippmMul_mm_32f(const Ipp32f* pSrc1, int src1Stride1, int
    src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, int src2Width, int src2Height, Ipp32f*
    pDst, int dstStride1, int dstStride2);
IppStatus ippmMul_mm_64f(const Ipp64f* pSrc1, int src1Stride1, int
    src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, int src2Width, int src2Height, Ipp64f*
    pDst, int dstStride1, int dstStride2);
IppStatus ippmMul_mm_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Width, int src2Height, Ipp32f** ppDst, int
    dstRoiShift);
IppStatus ippmMul_mm_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Width, int src2Height, Ipp64f** ppDst, int
    dstRoiShift);
```

Case 14: Transposed matrix - matrix operation

```
IppStatus ippmMul_tm_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride1, int src2Stride2, int src2Width, int src2Height,
    Ipp32f* pDst, int dstStride1, int dstStride2);
IppStatus ippmMul_tm_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride1, int src2Stride2, int src2Width, int src2Height,
    Ipp64f* pDst, int dstStride1, int dstStride2);
IppStatus ippmMul_tm_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Width, int src2Height, Ipp32f** ppDst, int
    dstRoiShift);
IppStatus ippmMul_tm_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Width, int src2Height, Ipp64f** ppDst, int
    dstRoiShift);
```

Case 15: Matrix - transposed matrix operation

```
IppStatus ippmMul_mt_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride1, int src2Stride2, int src2Width, int src2Height,
    Ipp32f* pDst, int dstStride1, int dstStride2);

IppStatus ippmMul_mt_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride1, int src2Stride2, int src2Width, int src2Height,
    Ipp64f* pDst, int dstStride1, int dstStride2);

IppStatus ippmMul_mt_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Width, int src2Height, Ipp32f** ppDst, int
    dstRoiShift);

IppStatus ippmMul_mt_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Width, int src2Height, Ipp64f** ppDst, int
    dstRoiShift);
```

Case 16: Transposed matrix - transposed matrix operation

```
IppStatus ippmMul_tt_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride1, int src2Stride2, int src2Width, int src2Height,
    Ipp32f* pDst, int dstStride1, int dstStride2);

IppStatus ippmMul_tt_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride1, int src2Stride2, int src2Width, int src2Height,
    Ipp64f* pDst, int dstStride1, int dstStride2);

IppStatus ippmMul_tt_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Width, int src2Height, Ipp32f** ppDst, int
    dstRoiShift);

IppStatus ippmMul_tt_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Width, int src2Height, Ipp64f** ppDst, int
    dstRoiShift);
```

Case 17: Matrix - matrix array operation

```
IppStatus ippmMul_mma_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride0, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);
```

```
IppStatus ippmMul_mma_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride0, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_mma_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Width, int src2Height,
    Ipp32f** ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mma_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Width, int src2Height,
    Ipp64f** ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mma_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride1, int src2Stride2, int
    src2Width, int src2Height, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int count);

IppStatus ippmMul_mma_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride1, int src2Stride2, int
    src2Width, int src2Height, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int count);
```

Case 18: Transposed matrix - matrix array operation

```
IppStatus ippmMul_tma_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride0, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_tma_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride0, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_tma_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Width, int src2Height,
    Ipp32f** ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tma_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Width, int src2Height,
    Ipp64f** ppDst, int dstRoiShift, int dstStride0, int count);
```

```
IppStatus ippmMul_tma_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride1, int src2Stride2, int
    src2Width, int src2Height, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int count);

IppStatus ippmMul_tma_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride1, int src2Stride2, int
    src2Width, int src2Height, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int count);
```

Case 19: Matrix - transposed matrix array operation

```
IppStatus ippmMul_mta_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride0, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_mta_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride0, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_mta_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Width, int src2Height,
    Ipp32f** ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mta_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Width, int src2Height,
    Ipp64f** ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mta_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride1, int src2Stride2, int
    src2Width, int src2Height, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int count);

IppStatus ippmMul_mta_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride1, int src2Stride2, int
    src2Width, int src2Height, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int count);
```

Case 20: Transposed matrix - transposed matrix array operation

```
IppStatus ippmMul_tta_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride0, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);
```

```
IppStatus ippmMul_tta_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride0, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_tta_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Width, int src2Height,
    Ipp32f** ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tta_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Width, int src2Height,
    Ipp64f** ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tta_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride1, int src2Stride2, int
    src2Width, int src2Height, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int count);

IppStatus ippmMul_tta_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride1, int src2Stride2, int
    src2Width, int src2Height, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int count);
```

Case 21: Matrix array - matrix operation

```
IppStatus ippmMul_mam_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_mam_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_mam_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Width, int src2Height, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mam_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Width, int src2Height, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int count);
```

```
IppStatus ippmMul_mam_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_mam_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int count);
```

Case 22: Transposed matrix array - matrix operation

```
IppStatus ippmMul_tam_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_tam_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_tam_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Width, int src2Height, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tam_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Width, int src2Height, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tam_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_tam_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int count);
```

Case 23: Matrix array - transposed matrix operation

```
IppStatus ippmMul_mat_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);
```

```
IppStatus ippmMul_mat_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_mat_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Width, int src2Height, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mat_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Width, int src2Height, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_mat_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_mat_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int count);
```

Case 24: Transposed matrix array - transposed matrix operation

```
IppStatus ippmMul_tat_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_tat_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f* pDst, int dstStride0, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_tat_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Width, int src2Height, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int count);

IppStatus ippmMul_tat_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Width, int src2Height, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int count);
```

```
IppStatus ippmMul_tat_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp32f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int count);

IppStatus ippmMul_tat_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride1, int src2Stride2, int src2Width,
    int src2Height, Ipp64f** ppDst, int dstRoiShift, int dstStride1,
    int dstStride2, int count);
```

Case 25: Matrix array - matrix array operation

```
IppStatus ippmMul_mama_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride0, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int count);

IppStatus ippmMul_mama_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride0, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int count);

IppStatus ippmMul_mama_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride0, int src2Width, int
    src2Height, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    count);

IppStatus ippmMul_mama_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride0, int src2Width, int
    src2Height, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    count);

IppStatus ippmMul_mama_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int count);

IppStatus ippmMul_mama_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int count);
```

Case 26: Transposed matrix array - matrix array operation

```
IppStatus ippmMul_tama_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride0, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int count);

IppStatus ippmMul_tama_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride0, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int count);

IppStatus ippmMul_tama_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride0, int src2Width, int
    src2Height, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    count);

IppStatus ippmMul_tama_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride0, int src2Width, int
    src2Height, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    count);

IppStatus ippmMul_tama_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int count);

IppStatus ippmMul_tama_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int count);
```

Case 27: Matrix array - transposed matrix array operation

```
IppStatus ippmMul_mata_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp32f* pSrc2, int src2Stride0, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int count);

IppStatus ippmMul_mata_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f* pSrc2, int src2Stride0, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int count);
```

IppStatus ippmMul_mata_32f_P(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride0*, int *src1Width*, int *src1Height*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, int *src2Width*, int *src2Height*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *count*);

IppStatus ippmMul_mata_64f_P(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int *src1Stride0*, int *src1Width*, int *src1Height*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, int *src2Width*, int *src2Height*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *count*);

IppStatus ippmMul_mata_32f_L(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, int *src1Width*, int *src1Height*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride1*, int *src2Stride2*, int *src2Width*, int *src2Height*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride1*, int *dstStride2*, int *count*);

IppStatus ippmMul_mata_64f_L(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, int *src1Width*, int *src1Height*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride1*, int *src2Stride2*, int *src2Width*, int *src2Height*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride1*, int *dstStride2*, int *count*);

Case 28: Transposed matrix array - transposed matrix array operation

IppStatus ippmMul_tata_32f(const Ipp32f* *pSrc1*, int *src1Stride0*, int *src1Stride1*, int *src1Stride2*, int *src1Width*, int *src1Height*, const Ipp32f* *pSrc2*, int *src2Stride0*, int *src2Stride1*, int *src2Stride2*, int *src2Width*, int *src2Height*, Ipp32f* *pDst*, int *dstStride0*, int *dstStride1*, int *dstStride2*, int *count*);

IppStatus ippmMul_tata_64f(const Ipp64f* *pSrc1*, int *src1Stride0*, int *src1Stride1*, int *src1Stride2*, int *src1Width*, int *src1Height*, const Ipp64f* *pSrc2*, int *src2Stride0*, int *src2Stride1*, int *src2Stride2*, int *src2Width*, int *src2Height*, Ipp64f* *pDst*, int *dstStride0*, int *dstStride1*, int *dstStride2*, int *count*);

IppStatus ippmMul_tata_32f_P(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride0*, int *src1Width*, int *src1Height*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, int *src2Width*, int *src2Height*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *count*);

IppStatus ippmMul_tata_64f_P(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int *src1Stride0*, int *src1Width*, int *src1Height*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, int *src2Width*, int *src2Height*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *count*);

IppStatus ippmMul_tata_32f_L(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, int *src1Width*, int *src1Height*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride1*, int *src2Stride2*, int *src2Width*, int *src2Height*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride1*, int *dstStride2*, int *count*);

```
IppStatus ippmMul_tata_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, int src1Width, int src1Height,
    const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride1, int
    src2Stride2, int src2Width, int src2Height, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int count);
```

Add

Adds matrix to another matrix.

Case 1: Matrix - matrix operation

```
IppStatus ippmAdd_mm_32f(const Ipp32f* pSrc1, int src1Stride1, int
    src1Stride2, const Ipp32f* pSrc2, int src2Stride1, int src2Stride2,
    Ipp32f* pDst, int dstStride1, int dstStride2, int width, int height);
IppStatus ippmAdd_mm_64f(const Ipp64f* pSrc1, int src1Stride1, int
    src1Stride2, const Ipp64f* pSrc2, int src2Stride1, int src2Stride2,
    Ipp64f* pDst, int dstStride1, int dstStride2, int width, int height);
IppStatus ippmAdd_mm_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f** ppDst, int
    dstRoiShift, int width, int height);
IppStatus ippmAdd_mm_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f** pDst, int
    dstRoiShift, int width, int height);
```

Case 2: Transposed matrix - matrix operation

```
IppStatus ippmAdd_tm_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp32f* pDst, int dstStride1, int dstStride2, int width,
    int height);
IppStatus ippmAdd_tm_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp64f* pDst, int dstStride1, int dstStride2, int width,
    int height);
IppStatus ippmAdd_tm_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f** ppDst, int
    dstRoiShift, int width, int height);
IppStatus ippmAdd_tm_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f** ppDst, int
    dstRoiShift, int width, int height);
```

Case 3: Transposed matrix - transposed matrix operation

```
IppStatus ippmAdd_tt_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp32f* pDst, int dstStride1, int dstStride2, int width,
    int height);
```

```
IppStatus ippmAdd_tt_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp64f* pDst, int dstStride1, int dstStride2, int width,
    int height);

IppStatus ippmAdd_tt_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f** ppDst, int
    dstRoiShift, int width, int height);

IppStatus ippmAdd_tt_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f** ppDst, int
    dstRoiShift, int width, int height);
```

Case 4: Matrix array - matrix operation

```
IppStatus ippmAdd_mam_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_mam_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_mam_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmAdd_mam_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmAdd_mam_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_mam_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 5: Transposed matrix array - matrix operation

```
IppStatus ippmAdd_tam_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_tam_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);
```

```
IppStatus ippmAdd_tam_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmAdd_tam_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmAdd_tam_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_tam_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 6: Matrix array - transposed matrix operation

```
IppStatus ippmAdd_mat_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_mat_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_mat_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmAdd_mat_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmAdd_mat_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_mat_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 7: Transposed matrix array - transposed matrix operation

```
IppStatus ippmAdd_tat_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_tat_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_tat_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmAdd_tat_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmAdd_tat_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmAdd_tat_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 8: Matrix array - matrix array operation

```
IppStatus ippmAdd_mama_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmAdd_mama_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmAdd_mama_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmAdd_mama_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);
```

```
IppStatus ippmAdd_mama_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmAdd_mama_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);
```

Case 9: Transposed matrix array - matrix array operation

```
IppStatus ippmAdd_tama_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmAdd_tama_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmAdd_tama_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmAdd_tama_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmAdd_tama_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmAdd_tama_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);
```

Case 10: Transposed matrix array - transposed matrix array operation

```
IppStatus ippmAdd_tata_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);
```

```
IppStatus ippmAdd_tata_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmAdd_tata_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmAdd_tata_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmAdd_tata_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmAdd_tata_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);
```

Sub

Subtracts matrix from another matrix.

Case 1: Matrix - matrix operation

```
IppStatus ippmSub_mm_32f(const Ipp32f* pSrc1, int src1Stride1, int
    src1Stride2, const Ipp32f* pSrc2, int src2Stride1, int src2Stride2,
    Ipp32f* pDst, int dstStride1, int dstStride2, int width, int height);

IppStatus ippmSub_mm_64f(const Ipp64f* pSrc1, int src1Stride1, int
    src1Stride2, const Ipp64f* pSrc2, int src2Stride1, int src2Stride2,
    Ipp64f* pDst, int dstStride1, int dstStride2, int width, int height);

IppStatus ippmSub_mm_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f** ppDst, int
    dstRoiShift, int width, int height);

IppStatus ippmSub_mm_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f** ppDst, int
    dstRoiShift, int width, int height);
```

Case 2: Transposed matrix - matrix operation

```
IppStatus ippmSub_tm_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp32f* pDst, int dstStride1, int dstStride2, int width,
    int height);
```

```
IppStatus ippmSub_tm_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp64f* pDst, int dstStride1, int dstStride2, int width,
    int height);

IppStatus ippmSub_tm_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f** ppDst, int
    dstRoiShift, int width, int height);

IppStatus ippmSub_tm_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f** ppDst, int
    dstRoiShift, int width, int height);
```

Case 3: Matrix - transposed matrix operation

```
IppStatus ippmSub_mt_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp32f* pDst, int dstStride1, int dstStride2, int width,
    int height);

IppStatus ippmSub_mt_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp64f* pDst, int dstStride1, int dstStride2, int width,
    int height);

IppStatus ippmSub_mt_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f** ppDst, int
    dstRoiShift, int width, int height);

IppStatus ippmSub_mt_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f** ppDst, int
    dstRoiShift, int width, int height);
```

Case 4: Transposed matrix - transposed matrix operation

```
IppStatus ippmSub_tt_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp32f* pDst, int dstStride1, int dstStride2, int width,
    int height);

IppStatus ippmSub_tt_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride1, int
    src2Stride2, Ipp64f* pDst, int dstStride1, int dstStride2, int width,
    int height);

IppStatus ippmSub_tt_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f** ppDst, int
    dstRoiShift, int width, int height);

IppStatus ippmSub_tt_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f** ppDst, int
    dstRoiShift, int width, int height);
```

Case 5: Matrix - matrix array operation

```
IppStatus ippmSub_mma_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride0, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mma_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride0, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mma_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_mma_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_mma_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mma_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 6: Transposed matrix - matrix array operation

```
IppStatus ippmSub_tma_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride0, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tma_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride0, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tma_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_tma_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);
```

```
IppStatus ippmSub_tma_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tma_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 7: Matrix - transposed matrix array operation

```
IppStatus ippmSub_mta_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride0, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mta_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride0, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mta_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_mta_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_mta_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mta_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 8: Transposed matrix - transposed matrix array operation

```
IppStatus ippmSub_tta_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f* pSrc2, int src2Stride0, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tta_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f* pSrc2, int src2Stride0, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);
```

```
IppStatus ippmSub_tta_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    const Ipp32f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_tta_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    const Ipp64f** ppSrc2, int src2RoiShift, int src2Stride0, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_tta_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tta_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 9: Matrix array - matrix operation

```
IppStatus ippmSub_mam_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mam_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mam_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_mam_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_mam_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mam_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 10: Transposed matrix array - matrix operation

```
IIppStatus ippmSub_tam_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tam_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tam_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_tam_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_tam_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tam_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 11: Matrix array - transposed matrix operation

```
IppStatus ippmSub_mat_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mat_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_mat_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_mat_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_mat_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

```
IppStatus ippmSub_mat_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 12: Transposed matrix array - transposed matrix operation

```
IppStatus ippmSub_tat_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tat_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tat_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, Ipp32f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_tat_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, Ipp64f**
    ppDst, int dstRoiShift, int dstStride0, int width, int height, int
    count);

IppStatus ippmSub_tat_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tat_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride1, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride1, int dstStride2, int width, int height, int count);
```

Case 13: Matrix array - matrix array operation

```
IppStatus ippmSub_mama_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_mama_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_mama_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);
```

```
IppStatus ippmSub_mama_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmSub_mama_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_mama_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);
```

Case 14: Transposed matrix array - matrix array operation

```
IppStatus ippmSub_tama_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_tama_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_tama_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmSub_tama_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmSub_tama_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_tama_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);
```

Case 15: Matrix array - transposed matrix array operation

```
IppStatus ippmSub_mata_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_mata_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp64f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_mata_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmSub_mata_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmSub_mata_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_mata_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);
```

Case 16: Transposed matrix array - transposed matrix array operation

```
IppStatus ippmSub_tata_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int
    src2Stride0, int src2Stride1, int src2Stride2, Ipp32f* pDst, int
    dstStride0, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_tata_64f(const Ipp64f* pSrc1, int src1Stride0, int
    src1Stride1, int src1Stride2, const Ipp64f* pSrc2, int src2Stride0,
    int src2Stride1, int src2Stride2, Ipp64f* pDst, int dstStride0,
    int dstStride1, int dstStride2, int width, int height, int count);

IppStatus ippmSub_tata_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp32f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp32f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);
```

```
IppStatus ippmSub_tata_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride0, const Ipp64f** ppSrc2, int src2RoiShift, int
    src2Stride0, Ipp64f** ppDst, int dstRoiShift, int dstStride0, int
    width, int height, int count);

IppStatus ippmSub_tata_32f_L(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);

IppStatus ippmSub_tata_64f_L(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Stride1, int src1Stride2, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride1, int src2Stride2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int width, int height,
    int count);
```

Gaxpy

Performs the “gaxpy” operation on a matrix.

Case 1: Matrix - vector operation

```
IppStatus ippmGaxpy_mv_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride2, int src2Len, const Ipp32f* pSrc3, int src2Stride3,
    int src3Len, Ipp32f* pDst, int dstStride2);

IppStatus ippmGaxpy_mv_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride2, int src2Len, const Ipp64f* pSrc3, int src3Stride2,
    int src3Len, Ipp64f* pDst, int dstStride2);

IppStatus ippmGaxpy_mv_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Len, const Ipp32f** ppSrc3, int src3RoiShift,
    int src3Len, Ipp32f** ppDst, int dstRoiShift);

IppStatus ippmGaxpy_mv_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Len, const Ipp64f** ppSrc3, int src3RoiShift,
    int src3Len, Ipp64f** ppDst, int dstRoiShift);
```

Case 2: Matrix - vector array operation

```
IppStatus ippmGaxpy_mva_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f* pSrc2,
    int src2Stride0, int src2Stride2, int src2Len, const Ipp32f* pSrc3,
    int src3Stride0, int src3Stride2, int src3Len, Ipp32f* pDst,
    int dstStride0, int dstStride2, int count);

IppStatus ippmGaxpy_mva_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f* pSrc2,
    int src2Stride0, int src2Stride2, int src2Len, const Ipp64f* pSrc3,
    int src3Stride0, int src3Stride2, int src3Len, Ipp64f* pDst,
    int dstStride0, int dstStride2, int count);
```

```
IppStatus ippmGaxpy_mva_32f_P(const Ipp32f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Len, const Ipp32f** ppSrc3,
    int src3RoiShift, int src3Stride0, int src3Len, Ipp32f** ppDst, int
    dstRoiShift, int dstStride0, int count);

IppStatus ippmGaxpy_mva_64f_P(const Ipp64f** ppSrc1, int src1RoiShift,
    int src1Width, int src1Height, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride0, int src2Len, const Ipp64f** ppSrc3,
    int src3RoiShift, int src3Stride0, int src3Len, Ipp64f** ppDst, int
    dstRoiShift, int dstStride0, int count);

IppStatus ippmGaxpy_mva_32f_L(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride2, int src2Len, const
    Ipp32f** ppSrc3, int src3RoiShift, int src3Stride2, int src3Len,
    Ipp32f** ppDst, int dstRoiShift, int dstStride2, int count);

IppStatus ippmGaxpy_mva_64f_L(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int src1Width, int src1Height, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride2, int src2Len, const
    Ipp64f** ppSrc3, int src3RoiShift, int src3Stride2, int sr32Len,
    Ipp64f** ppDst, int dstRoiShift, int dstStride2, int count;)
```

Linear System Solution Functions

LUDecomp

Decomposes square matrix into product of upper and lower triangular matrices.

Case 1: Matrix operation

```
IppStatus ippmLUDecomp_m_32f(const Ipp32f* pSrc, int srcStride1,
    int srcStride2, int* pDstIndex, Ipp32f* pDst, int dstStride1,
    int dstStride2, int widthHeight);

IppStatus ippmLUDecomp_m_64f(const Ipp64f* pSrc, int srcStride1,
    int srcStride2, int* pDstIndex, Ipp64f* pDst, int dstStride1,
    int dstStride2, int widthHeight);

IppStatus ippmLUDecomp_m_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int* pDstIndex, Ipp32f** ppDst, int dstRoiShift, int widthHeight);

IppStatus ippmLUDecomp_m_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int* pDstIndex, Ipp64f** ppDst, int dstRoiShift, int widthHeight);
```

Case 2: Matrix array operation

```
IppStatus ippmLUDecomp_ma_32f(const Ipp32f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, int* pDstIndex, Ipp32f* pDst,
    int dstStride0, int dstStride1, int dstStride2, int widthHeight, int
    count);
```

```
IppStatus ippmLUDecomp_ma_64f(const Ipp64f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, int* pDstIndex, Ipp64f* pDst,
    int dstStride0, int dstStride1, int dstStride2, int widthHeight, int
    count);

IppStatus ippmLUDecomp_ma_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, int* pDstIndex, Ipp32f** ppDst, int dstRoiShift,
    int dstStride0, int widthHeight, int count);

IppStatus ippmLUDecomp_ma_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, int* pDstIndex, Ipp64f** ppDst, int dstRoiShift,
    int dstStride0, int widthHeight, int count);

IppStatus ippmLUDecomp_ma_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, int* pDstIndex, Ipp32f** ppDst,
    int dstRoiShift, int dstStride1, int dstStride2, int widthHeight,
    int count);

IppStatus ippmLUDecomp_ma_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, int* pDstIndex, Ipp64f** ppDst,
    int dstRoiShift, int dstStride1, int dstStride2, int widthHeight,
    int count);
```

LUBackSubst

Solves system of linear equations with LU-factored square matrix.

Case 1: Matrix - vector operation

```
IppStatus ippmLUBackSubst_mv_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, int* pSrcIndex, const Ipp32f* pSrc2, int
    src2Stride2, Ipp32f* pDst, int dstStride2, int widthHeight);

IppStatus ippmLUBackSubst_mv_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, int* pSrcIndex, const Ipp64f* pSrc2, int
    src2Stride2, Ipp64f* pDst, int dstStride2, int widthHeight);

IppStatus ippmLUBackSubst_mv_32f_P(const Ipp32f** ppSrc1, int
    src1RoiShift, int* pSrcIndex, const Ipp32f** ppSrc2, int
    src2RoiShift, Ipp32f** ppDst, int dstRoiShift, int widthHeight);

IppStatus ippmLUBackSubst_mv_64f_P(const Ipp64f** ppSrc1, int
    src1RoiShift, int* pSrcIndex, const Ipp64f** ppSrc2, int
    src2RoiShift, Ipp64f** ppDst, int dstRoiShift, int widthHeight);
```

Case 2: Matrix - vector array operation

```
IppStatus ippmLUBackSubst_mva_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int* pSrcIndex, const Ipp32f*
    pSrc2, int src2Stride0, int src2Stride2, Ipp32f* pDst, int
    dstStride0, int dstStride2, int widthHeight, int count);

IppStatus ippmLUBackSubst_mva_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, int* pSrcIndex, const Ipp64f*
    pSrc2, int src2Stride0, int src2Stride2, Ipp64f* pDst, int
    dstStride0, int dstStride2, int widthHeight, int count);
```

IppStatus ippmLUBackSubst_mva_32f_P(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int* *pSrcIndex*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmLUBackSubst_mva_64f_P(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int* *pSrcIndex*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *widthHeight*, int *count*);

IppStatus ippmLUBackSubst_mva_32f_L(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, int* *pSrcIndex*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmLUBackSubst_mva_64f_L(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, int* *pSrcIndex*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *widthHeight*, int *count*);

Case 3: Matrix array - vector array operation

IppStatus ippmLUBackSubst_mava_32f(const Ipp32f* *pSrc1*, int *src1Stride0*, int *src1Stride1*, int *src1Stride2*, int* *pSrcIndex*, const Ipp32f* *pSrc2*, int *src2Stride0*, int *src2Stride2*, Ipp32f* *pDst*, int *dstStride0*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmLUBackSubst_mava_64f(const Ipp64f* *pSrc1*, int *src1Stride0*, int *src1Stride1*, int *src1Stride2*, int* *pSrcIndex*, const Ipp64f* *pSrc2*, int *src2Stride0*, int *src2Stride2*, Ipp64f* *pDst*, int *dstStride0*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmLUBackSubst_mava_32f_P(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride0*, int* *pSrcIndex*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *widthHeight*, int *count*);

IppStatus ippmLUBackSubst_mava_64f_P(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int *src1Stride0*, int* *pSrcIndex*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *widthHeight*, int *count*);

IppStatus ippmLUBackSubst_mava_32f_L(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, int* *pSrcIndex*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmLUBackSubst_mava_64f_L(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, int* *pSrcIndex*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *widthHeight*, int *count*);

CholeskyDecomp

Performs Cholesky decomposition of a symmetric positive definite square matrix.

Case 1: Matrix operation

```
IppStatus ippmCholeskyDecomp_m_32f(const Ipp32f* pSrc, int srcStride1,
    int srcStride2, Ipp32f* pDst, int dstStride1, int dstStride2,
    int widthHeight);
IppStatus ippmCholeskyDecomp_m_64f(const Ipp64f* pSrc, int srcStride1,
    int srcStride2, Ipp64f* pDst, int dstStride1, int dstStride2,
    int widthHeight);
IppStatus ippmCholeskyDecomp_m_32f_P(const Ipp32f** ppSrc, int
    srcRoiShift, Ipp32f** ppDst, int dstRoiShift, int widthHeight);
IppStatus ippmCholeskyDecomp_m_64f_P(const Ipp64f** ppSrc, int
    srcRoiShift, Ipp64f** ppDst, int dstRoiShift, int widthHeight);
```

Case 2: Matrix array operation

```
IppStatus ippmCholeskyDecomp_ma_32f(const Ipp32f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, Ipp32f* pDst, int dstStride0,
    int dstStride1, int dstStride2, int widthHeight, int count);
IppStatus ippmCholeskyDecomp_ma_64f(const Ipp64f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, Ipp64f* pDst, int dstStride0,
    int dstStride1, int dstStride2, int widthHeight, int count);
IppStatus ippmCholeskyDecomp_ma_32f_P(const Ipp32f** ppSrc, int
    srcRoiShift, int srcStride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride2, int widthHeight, int count);
IppStatus ippmCholeskyDecomp_ma_64f_P(const Ipp64f** ppSrc, int
    srcRoiShift, int srcStride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride2, int widthHeight, int count);
IppStatus ippmCholeskyDecomp_ma_32f_L(const Ipp32f** ppSrc, int
    srcRoiShift, int srcStride1, int srcStride2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int widthHeight, int
    count);
IppStatus ippmCholeskyDecomp_ma_64f_L(const Ipp64f** ppSrc, int
    srcRoiShift, int srcStride1, int srcStride2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride1, int dstStride2, int widthHeight, int
    count);
```

CholeskyBackSubst

Solves system of linear equations using the Cholesky triangular factor.

Case 1: Matrix -vector operation

```
IppStatus ippmCholeskyBackSubst_mv_32f(const Ipp32f* pSrc1, int
    src1Stride1, int src1Stride2, const Ipp32f* pSrc2, int src2Stride2,
    Ipp32f* pDst, int dstStride2, int widthHeight);
```

IppStatus ippmCholeskyBackSubst_mv_64f(const Ipp64f* *pSrc1*, int *src1Stride1*, int *src1Stride2*, const Ipp64f* *pSrc2*, int *src2Stride2*, Ipp64f* *pDst*, int *dstStride2*, int *widthHeight*);

IppStatus ippmCholeskyBackSubst_mv_32f_P(const Ipp32f** *ppSrc1*, int *src1RoiShift*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, Ipp32f** *ppDst*, int *dstRoiShift*, int *widthHeight*);

IppStatus ippmCholeskyBackSubst_mv_64f_P(const Ipp64f** *ppSrc1*, int *src1RoiShift*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, Ipp64f** *ppDst*, int *dstRoiShift*, int *widthHeight*);

Case 2: Matrix - vector array operation

IppStatus ippmCholeskyBackSubst_mva_32f(const Ipp32f* *pSrc1*, int *src1Stride1*, int *src1Stride2*, const Ipp32f* *pSrc2*, int *src2Stride0*, int *src2Stride2*, Ipp32f* *pDst*, int *dstStride0*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmCholeskyBackSubst_mva_64f(const Ipp64f* *pSrc1*, int *src1Stride1*, int *src1Stride2*, const Ipp64f* *pSrc2*, int *src2Stride0*, int *src2Stride2*, Ipp64f* *pDst*, int *dstStride0*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmCholeskyBackSubst_mva_32f_P(const Ipp32f** *ppSrc1*, int *src1RoiShift*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmCholeskyBackSubst_mva_64f_P(const Ipp64f** *ppSrc1*, int *src1RoiShift*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmCholeskyBackSubst_mva_32f_L(const Ipp32f* *pSrc1*, int *src1Stride1*, int *src1Stride2*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmCholeskyBackSubst_mva_64f_L(const Ipp64f* *pSrc1*, int *src1Stride1*, int *src1Stride2*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *widthHeight*, int *count*);

Case 3: Matrix array - vector array operation

IppStatus ippmCholeskyBackSubst_mava_32f(const Ipp32f* *pSrc1*, int *src1Stride0*, int *src1Stride1*, int *src1Stride2*, const Ipp32f* *pSrc2*, int *src2Stride0*, int *src2Stride2*, Ipp32f* *pDst*, int *dstStride0*, int *dstStride2*, int *widthHeight*, int *count*);

IppStatus ippmCholeskyBackSubst_mava_64f(const Ipp64f* *pSrc1*, int *src1Stride0*, int *src1Stride1*, int *src1Stride2*, const Ipp64f* *pSrc2*, int *src2Stride0*, int *src2Stride2*, Ipp64f* *pDst*, int *dstStride0*, int *dstStride2*, int *widthHeight*, int *count*);

```
IppStatus ippmCholeskyBackSubst_mava_32f_P(const Ipp32f** ppSrc1,
    int src1RoiShift, int src1Stride2, const Ipp32f** ppSrc2, int
    src2RoiShift, int src2Stride2, Ipp32f** ppDst, int dstRoiShift, int
    dstStride2, int widthHeight, int count);
IppStatus ippmCholeskyBackSubst_mava_64f_P(const Ipp64f** ppSrc1,
    int src1RoiShift, int src1Stride2, const Ipp64f** ppSrc2, int
    src2RoiShift, int src2Stride2, Ipp64f** ppDst, int dstRoiShift, int
    dstStride2, int widthHeight, int count);
IppStatus ippmCholeskyBackSubst_mava_32f_L(const Ipp32f** ppSrc1,
    int src1RoiShift, int src1Stride1, int src1Stride2, const Ipp32f**
    ppSrc2, int src2RoiShift, int src2Stride2, Ipp32f** ppDst, int
    dstRoiShift, int dstStride2, int widthHeight, int count);
IppStatus ippmCholeskyBackSubst_mava_64f_L(const Ipp64f** ppSrc1,
    int src1RoiShift, int src1Stride1, int src1Stride2, const Ipp64f**
    ppSrc2, int src2RoiShift, int src2Stride2, Ipp64f** ppDst, int
    dstRoiShift, int dstStride2, int widthHeight, int count);
```

Least Squares Problem Functions

QRDecomp

Computes the QR decomposition for the given matrix.

Case 1: Matrix operation

```
IppStatus ippmQRDecomp_m_32f(const Ipp32f* pSrc, int srcStride1,
    int srcStride2, Ipp32f* pBuffer, Ipp32f* pDst, int dstStride1,
    int dstStride2, int width, int height);
IppStatus ippmQRDecomp_m_64f(const Ipp64f* pSrc, int srcStride1,
    int srcStride2, Ipp64f* pBuffer, Ipp64f* pDst, int dstStride1,
    int dstStride2, int width, int height);
IppStatus ippmQRDecomp_m_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    Ipp32f* pBuffer, Ipp32f** ppDst, int dstRoiShift, int width, int
    height);
IppStatus ippmQRDecomp_m_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    Ipp64f* pBuffer, Ipp64f** ppDst, int dstRoiShift, int width, int
    height);
```

Case 2: Matrix array operation

```
IppStatus ippmQRDecomp_ma_32f(const Ipp32f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, Ipp32f* pBuffer, Ipp32f* pDst,
    int dstStride0, int dstStride1, int dstStride2, int width, int
    height, int count);
IppStatus ippmQRDecomp_ma_64f(const Ipp64f* pSrc, int srcStride0,
    int srcStride1, int srcStride2, Ipp64f* pBuffer, Ipp64f* pDst,
    int dstStride0, int dstStride1, int dstStride2, int width, int
    height, int count);
```

```
IppStatus ippmQRDecomp_ma_32f_P(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp32f* pBuffer, Ipp32f** ppDst, int dstRoiShift,
    int dstStride0, int width, int height, int count);

IppStatus ippmQRDecomp_ma_64f_P(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride0, Ipp64f* pBuffer, Ipp64f** ppDst, int dstRoiShift,
    int dstStride0, int width, int height, int count);

IppStatus ippmQRDecomp_ma_32f_L(const Ipp32f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, Ipp32f* pBuffer, Ipp32f** ppDst,
    int dstRoiShift, int dstStride1, int dstStride2, int width, int
    height, int count);

IppStatus ippmQRDecomp_ma_64f_L(const Ipp64f** ppSrc, int srcRoiShift,
    int srcStride1, int srcStride2, Ipp64f* pBuffer, Ipp64f** ppDst,
    int dstRoiShift, int dstStride1, int dstStride2, int width, int
    height, int count);
```

QRBackSubst

Solves least squares problem for QR-decomposed matrix.

Case 1: Matrix - vector operation

```
IppStatus ippmQRBackSubst_mv_32f(const Ipp32f* pSrc1, int src1Stride1,
    int src1Stride2, Ipp32f* pBuffer, const Ipp32f* pSrc2, int
    src2Stride2, Ipp32f* pDst, int dstStride2, int width, int height);

IppStatus ippmQRBackSubst_mv_64f(const Ipp64f* pSrc1, int src1Stride1,
    int src1Stride2, Ipp64f* pBuffer, const Ipp64f* pSrc2, int
    src2Stride2, Ipp64f* pDst, int dstStride2, int width, int height);

IppStatus ippmQRBackSubst_mv_32f_P(const Ipp32f** ppSrc1, int
    src1RoiShift, Ipp32f* pBuffer, const Ipp32f** ppSrc2, int
    src2RoiShift, Ipp32f** ppDst, int dstRoiShift, int width, int
    height);

IppStatus ippmQRBackSubst_mv_64f_P(const Ipp64f** ppSrc1, int
    src1RoiShift, Ipp64f* pBuffer, const Ipp64f** ppSrc2, int
    src2RoiShift, Ipp64f** ppDst, int dstRoiShift, int width, int
    height);
```

Case 2: Matrix - vector array operation

```
IppStatus ippmQRBackSubst_mva_32f(const Ipp32f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, Ipp32f* pBuffer, const Ipp32f*
    pSrc2, int src2Stride0, int src2Stride2, Ipp32f* pDst, int
    dstStride0, int dstStride2, int width, int height, int count);

IppStatus ippmQRBackSubst_mva_64f(const Ipp64f* pSrc1, int src1Stride0,
    int src1Stride1, int src1Stride2, Ipp64f* pBuffer, const Ipp64f*
    pSrc2, int src2Stride0, int src2Stride2, Ipp64f* pDst, int
    dstStride0, int dstStride2, int width, int height, int count);
```

IppStatus ippmQRBackSubst_mva_32f_P(const Ipp32f** *ppSrc1*, int *src1RoiShift*, Ipp32f* *pBuffer*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *width*, int *height*, int *count*);

IppStatus ippmQRBackSubst_mva_64f_P(const Ipp64f** *ppSrc1*, int *src1RoiShift*, Ipp64f* *pBuffer*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *width*, int *height*, int *count*);

IppStatus ippmQRBackSubst_mva_32f_L(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, Ipp32f* *pBuffer*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmQRBackSubst_mva_64f_L(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, Ipp64f* *pBuffer*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *width*, int *height*, int *count*);

Case 3: Matrix array - vector array operation

IppStatus ippmQRBackSubst_mava_32f(const Ipp32f* *pSrc1*, int *src1Stride0*, int *src1Stride1*, int *src1Stride2*, Ipp32f* *pBuffer*, const Ipp32f* *pSrc2*, int *src2Stride0*, int *src2Stride2*, Ipp32f* *pDst*, int *dstStride0*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmQRBackSubst_mava_64f(const Ipp64f* *pSrc1*, int *src1Stride0*, int *src1Stride1*, int *src1Stride2*, Ipp64f* *pBuffer*, const Ipp64f* *pSrc2*, int *src2Stride0*, int *src2Stride2*, Ipp64f* *pDst*, int *dstStride0*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmQRBackSubst_mava_32f_P(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride0*, Ipp32f* *pBuffer*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *width*, int *height*, int *count*);

IppStatus ippmQRBackSubst_mava_64f_P(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int *src1Stride0*, Ipp64f* *pBuffer*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride0*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride0*, int *width*, int *height*, int *count*);

IppStatus ippmQRBackSubst_mava_32f_L(const Ipp32f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, Ipp32f* *pBuffer*, const Ipp32f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp32f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *width*, int *height*, int *count*);

IppStatus ippmQRBackSubst_mava_64f_L(const Ipp64f** *ppSrc1*, int *src1RoiShift*, int *src1Stride1*, int *src1Stride2*, Ipp64f* *pBuffer*, const Ipp64f** *ppSrc2*, int *src2RoiShift*, int *src2Stride2*, Ipp64f** *ppDst*, int *dstRoiShift*, int *dstStride2*, int *width*, int *height*, int *count*);

Eigenvalue Problem Functions

EigenValuesVectorsSym

Finds eigenvalues and eigenvectors for real symmetric matrices (solves symmetric eigenvalue problem).

Case 1: Matrix operation

```
IppStatus ippmEigenValuesVectorsSym_m_32f (const Ipp32f* pSrc, int
    srcStride1, int  srcStride2, Ipp32f* pBuffer, Ipp32f* pDstVectors,
    int dstStride1, int dstStride2, Ipp32f* pDstValues, int widthHeight);

IppStatus ippmEigenValuesVectorsSym_m_64f (const Ipp64f* pSrc, int
    srcStride1, int  srcStride2, Ipp64f* pBuffer,  Ipp64f* pDstVectors,
    int dstStride1, int dstStride2, Ipp64f* pDstValues, int widthHeight);

IppStatus ippmEigenValuesVectorsSym_m_32f_P (const Ipp32f** ppSrc, int
    srcRoiShift, Ipp32f* pBuffer, Ipp32f** ppDstVectors, int dstRoiShift,
    Ipp32f* pDstValues, int widthHeight);

IppStatus ippmEigenValuesVectorsSym_m_64f_P (const Ipp64f** ppSrc, int
    srcRoiShift, Ipp64f* pBuffer, Ipp64f** ppDstVectors, int dstRoiShift,
    Ipp64f* pDstValues, int widthHeight);
```

Case 2: Matrix array operation

```
IppStatus ippmEigenValuesVectorsSym_ma_32f (const Ipp32f* pSrc, int
    srcStride0, int srcStride1, int srcStride2, Ipp32f* pBuffer, Ipp32f*
    pDstVectors, int dstStride0, int dstStride1, int dstStride2, Ipp32f*
    pDstValues, int widthHeight, int count);

IppStatus ippmEigenValuesVectorsSym_ma_32f_P (const Ipp32f** ppSrc, int
    srcRoiShift, int srcStride0, Ipp32f* pBuffer, Ipp32f** ppDstVectors,
    int dstRoiShift, int dstStride0, Ipp32f* pDstValues, int widthHeight,
    int count);

IppStatus ippmEigenValuesVectorsSym_ma_32f_L (const Ipp32f** ppSrc, int
    srcRoiShift, int srcStride1,  int srcStride2, Ipp32f* pBuffer,
    Ipp32f** ppDstVectors,  int dstRoiShift, int dstStride1, int
    dstStride2, Ipp32f* pDstValues, int widthHeight, int count);

IppStatus ippmEigenValuesVectorsSym_ma_64f (const Ipp64f* pSrc, int
    srcStride0, int srcStride1, int  srcStride2, Ipp64f* pBuffer, Ipp64f*
    pDstVectors, int dstStride0, int dstStride1, int dstStride2, Ipp64f*
    pDstValues, int widthHeight, int count);

IppStatus ippmEigenValuesVectorsSym_ma_64f_P (const Ipp64f** ppSrc, int
    srcRoiShift, int srcStride0, Ipp64f* pBuffer, Ipp64f** ppDstVectors,
    int dstRoiShift, int dstStride0, Ipp64f* pDstValues, int widthHeight,
    int count);

IppStatus ippmEigenValuesVectorsSym_ma_64f_L (const Ipp64f** ppSrc, int
    srcRoiShift, int srcStride1, int srcStride2, Ipp64f* pBuffer,
    Ipp64f** ppDstVectors, int dstRoiShift, int dstStride1, int
    dstStride2, Ipp64f* pDstValues, int widthHeight, int count);
```

EigenValuesSym

Find eigenvalues for real symmetric matrices.

Case 1: Matrix operation

```
IppStatus ippmEigenValuesSym_m_32f (const Ipp32f* pSrc, int srcStride1,
    int  srcStride2,  Ipp32f* pBuffer, Ipp32f* pDstValues, int
    widthHeight);

IppStatus ippmEigenValuesSym_m_64f (const Ipp64f* pSrc, int srcStride1,
    int srcStride2, Ipp64f* pBuffer, Ipp64f* pDstValues, int
    widthHeight);

IppStatus ippmEigenValuesSym_m_32f_P (const Ipp32f** ppSrc, int
    srcRoiShift, Ipp32f* pBuffer, Ipp32f* pDstValues, int widthHeight);

IppStatus ippmEigenValuesSym_m_64f_P (const Ipp64f** ppSrc,  int
    srcRoiShift, Ipp64f* pBuffer, Ipp64f* pDstValues, int widthHeight);
```

Case 2: Matrix array operation

```
IppStatus ippmEigenValuesSym_ma_32f (const Ipp32f* pSrc, int srcStride0,
    int srcStride1, int  srcStride2, Ipp32f* pBuffer, Ipp32f* pDstValues,
    int widthHeight, int count);

IppStatus ippmEigenValuesSym_ma_32f_P (const Ipp32f** ppSrc,  int
    srcRoiShift, int srcStride0, Ipp32f* pBuffer, Ipp32f* pDstValues,
    int widthHeight, int count);

IppStatus ippmEigenValuesSym_ma_32f_L (const Ipp32f** ppSrc, int
    srcRoiShift, int srcStride1,  int  srcStride2,  Ipp32f* pBuffer,
    Ipp32f* pDstValues, int widthHeight, int count);

IppStatus ippmEigenValuesSym_ma_64f (const Ipp64f* pSrc, int srcStride0,
    int srcStride1,  int  srcStride2,  Ipp64f* pBuffer,  Ipp64f*
    pDstValues, int widthHeight, int count);

IppStatus ippmEigenValuesSym_ma_64f_P (const Ipp64f** ppSrc, int
    srcRoiShift, int srcStride0,  Ipp64f* pBuffer, Ipp64f* pDstValues,
    int widthHeight, int count);

IppStatus ippmEigenValuesSym_ma_64f_L (const Ipp64f** ppSrc, int
    srcRoiShift, int srcStride1,  int  srcStride2,  Ipp64f* pBuffer,
    Ipp64f* pDstValues, int widthHeight, int count);
```